**PROFESSIONAL INDUSTRIAL TRAINING**

**SUBMITTED TO**

**Department of Computer Science**

**And Engineering**

**Model Institute of Engineering and Technology (Autonomous)**

**Jammu, India**

**/ INTERNSHIP REPORT**

**ON**

“Internet of Things”

## AT

**Solitaire Infosys**

**Pvt. Ltd.**

**C-110, Industrial Area Phase-VII, Mohali, India.**

## AN INDUSTRY INTERNSHIP REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

**FOR THE AWARD OF DEGREE OF**

# BACHELOR OF ENGINEERING

**In**

# Department of Computer Science and Engineering

**SUBMITTED BY**

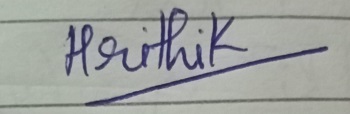
Hrithik Sharma

Roll Number: (2020a1r034)



**CANDIDATES’ DECLARATION**

I, **Hrithik Sharma, Roll Number (2020a1r034)** hereby declare that the work which is being presented in the Industry Internship Report entitled, “**Internet of Things”** in partial fulfillment of requirement for the award of degree of B.E. (Computer Science) and submitted in the Department of Computer Science And Engineering, Model Institute of Engineering and Technology (Autonomous), Jammu is an authentic record of my own work carried by me at “Solitaire Infosys Pvt. Ltd. Mohali” under the supervision and mentorship of **Ms. Veena** (Software Developer at Solitaire Infosys PVT. LTD). The matter presented in this report has not been submitted in this or any other University / Institute for the award of B.E. Degree.



*Signature of the Student Dated*:

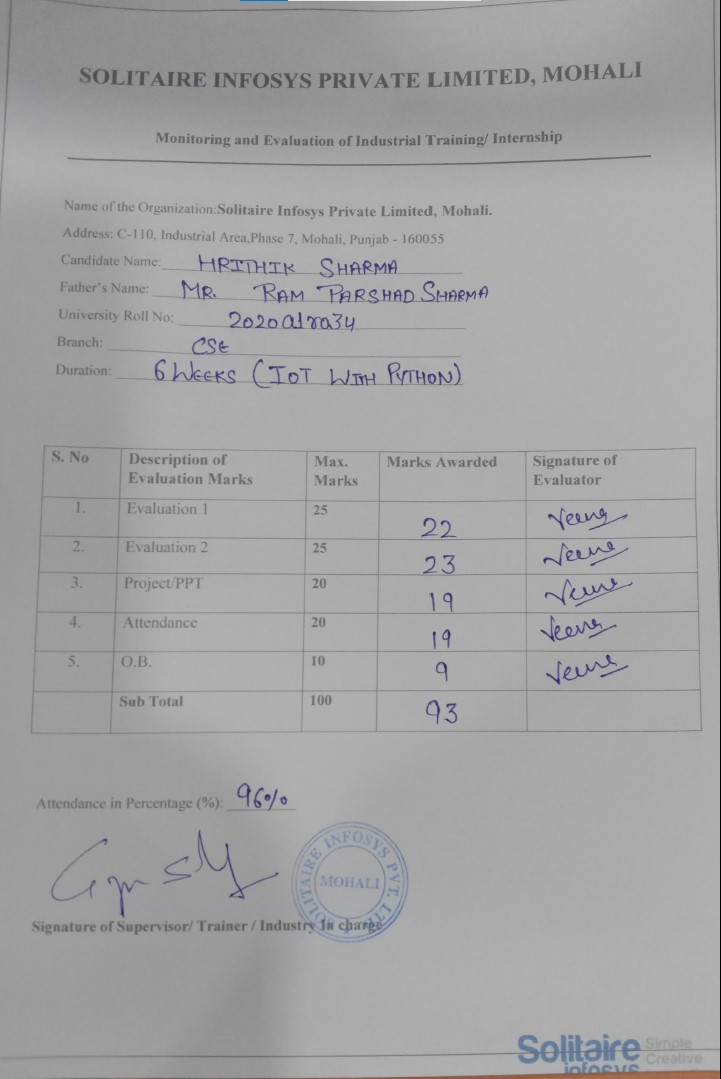
## (Hrithik Sharma) 3RD Nov 2022

**2020a1r034**

**Internship Certificate**



**Supervisor Evaluation of Intern**



**Self-Evaluation**

During the internship program we have got the opportunity to

learn and gain the skill. In this internship we learn about the

programming languages like Embedded C++ and Python.

We also got to know about the one of the important emerging

modern world technologies like IOT (Internet of Things).

In python, we learn about the concepts like data types, arrays,

Python libraries, input, output, built in functions, user-defined

functions, Loops like for loop and while loops, conditional statements

(if-else).

In IOT, we learn about various sensors like temperature sensors and

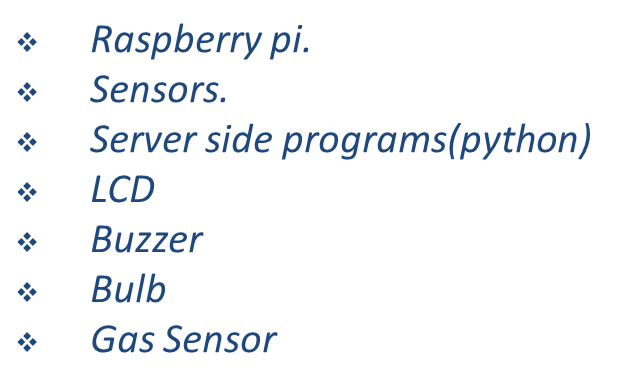
Raspberry pi.

We make a final project “HOME AUTOMATION SYSTEM”

based on the learnings we have done in this internship program

and using python programs.

In this Project, we used the following components:-



Conclusion:-

We learn some of new technologies and gain a overall good experience.

**Department Of Computer Science And Engineering**

**Model Institute of Engineering and Technology (Autonomous)**

**Kot Bhalwal, Jammu, India**

***(NAAC “A” Grade Accredited)***

**Ref. No.: 2020a1r034 Date:03RD Nov 2022**

# CERTIFICATE

Certified that this Industry Internship Report entitled **“Internet of Things”** is the bonafide work of “**Hrithik Sharma, Roll No. 2020a1r034, of 5th Semester, Department of Computer Science And Engineering, Model Institute of Engineering and Technology (Autonomous), Jammu”,** who carried out the Industry Internship at **“Solitaire Infosys Pvt. Ltd. Mohali”** work under my mentorship during 13th July, 2022 to 27th August, 2022.

## (Ms. Mekhla Sharma)

**Mentor-Internal Supervisor**

**College Mentor’s Designation**

## Department of Computer Science and Engineering, MIET

*This is to certify that the above statement is correct to the best of my knowledge.*

# ACKNOWLEDGEMENTS

This Summer internship opportunity was a great chance for learning and professional development. I am grateful for having a chance to meet so many wonderful people and professionals who led me though this internship period.

It is my pleasant duty to pay my heartfelt gratitude to Ms. Veena Sharma, Software Engineer Solitaire Infosys PVT. LTD who have guided me through the course of this Internship.

I must record my deep sense of gratitude to Prof. (Dr.) Ankur Gupta (Director, MIET) and Prof. (Dr.) Ashok Kumar (Dean Academics & HOD CSE, MIET) for their guidance, constant inspiration and encouragement, and for their keen involvement throughout the course of present work.

Gratitude and thanks although mean a very small thing to convey my thanks to my parents who have always given me a parental source of love, motivation and strength right from the journey of my life.

Bearing in mind previous I am using this opportunity to express my deepest gratitude and special thanks to the teachers who in spite of being extraordinarily busy with their duties, took time out to hear, guide and keep me on the correct path and allowing me to carry out my project at their esteemed organization and extending during the training.

I perceive this opportunity as a big milestone in my career development. I will strive to use gained skills and knowledge in the best possible way, and I will continue to work on their improvement, in order to attain desired career objectives. Hope to continue cooperation with all of you in the future.

I express my sincere gratitude to Solitaire Infosys Pvt. Ltd, Mohali and Model Institute of Engineering and Technology (Autonomous), Jammu for giving me the opportunity.

**Hrithik Sharma**

**2020a1r034**

|  |  |  |
| --- | --- | --- |
|  | **CONTENTS** |  |
|  | Page No. |
| **CHAPTER-1** | **INTRODUCTION TO IoT** | **12** |
|  | 1.1 INTRODUCTION | 13 |
|  | 1.2 HISTORY OF IoT | 13 |
|  | 1.3 Major Components Used In IoT  1.3.1 Types Of Sensors  1.4 Control Units  1.4.1 Cloud Computing  1.4.2 Availability of big data  1.4.3 Networking Connection  1.5 IoT Enablers  1.6 Working with IoT  1.7 Applications | 14  15  15  16  16  16  17  18  19 |
| **CHAPTER 2** | **Architecture and Domains** |  |
|  | 2.1 IoT Architecture | 20 |
|  | 2.1.1 Stages of IoT Solution Architecture | 22 |
|  | 2.2 Domains in IoT  2.2.1 IoT in Smart home applications  2.2.2 Security System and its working | 25 |
|  | 2.3 Technology Involved In IoT | 28 |
|  | 2.3.1 Internet/Web and Networking Basics  OSI Model | 29 |
|  | 2.3.2 Networking Standards and Technologies | 30 |
|  | 2.3.3 Layers Used In IoT | 32 |
|  | 2.3.4 Network Access and Physical Layer | 33 |
|  | 2.3.5 Application Layer | 35 |
|  | 2.4 IoT Networking Considerations and Challenges | 36 |
|  | 2.5 Factors In Shaping a secure network in IoT | 38 |

|  |  |  |
| --- | --- | --- |
| **CHAPTER 3** | **IoT Devices** | **39** |
|  | 3.1 INTRODUCTION | 40 |
|  | 3.1.1 Examples | 41 |
|  | 3.2 How IoT Devices Work | 42 |
|  | 3.3 IoT device Management | 43 |
|  | 3.4 Connectivity and Networking | 44 |
|  | 3.5 Security Risks  3.6 IoT Devices Trends and Growth  3.7 Growth Graph | 45  46 |
| **CHAPTER4** | **Sensors** | **47** |
| **CHAPTER 5**  **CHAPTER 6**  **REFERENCES** | 4.1 What are IoT Sensors?  4.2 Types and Uses of Sensors  4.3 Sensors help drive the value of IoT  4.4 Conclusion  **Python**  5.1 Introduction  5.2 Source Code  5.3 User Defined Function  5.4 Indentation  5.5 Code Checking  5.6 Variables  5.7 Features of Python  5.8 Libraries    **Project – Home Automation System**  6.1 Objective  6.2 Scope  6.3 System Requirement  6.4 Technology Used  6.5 Raspberry PI  6.6 Workflow  6.7 Source Code  6.8 Model  6.9 Benefit  6.10 Conclusion | 48  49  50  51  53  55  57  59  61  62  63  64  64  65  66  66  67  68  69  70  71  72 |

|  |  |  |
| --- | --- | --- |
| **Fig No.** |  | **Page No.** |
| Figure 1.1 |  | 17 |
| Figure 2.1 |  | 21 |
| Figure 2.2 |  | 22 |
| Figure 2.3 |  | 23 |
| Figure 2.4 |  | 24 |
| Figure 2.5 |  | 25 |
| Figure 3.1 |  | 40 |
| Figure 3.2 |  | 41 |
| Figure 3.3 |  | 42 |
| Figure 3.4 |  | 47 |
| Figure 4.1 |  | 48 |
| Figure 5.1 |  | 64 |
| Figure 6.1 |  | 65 |
| Figure 6.2 |  | 68 |
| Figure 6.3 |  | 68 |
| Figure 6.4 |  | 69 |

**CHAPTER 1 INTRODUCTION TO IoT**

# INTRODUCTION

# Internet of Things (IoT) is the networking of physical objects that contain electronics embedded within their architecture in order to communicate and sense interactions amongst each other or with respect to the external environment. In the upcoming years, IoT-based technology will offer advanced levels of services and practically change the way people lead their daily lives. Advancements in medicine, power, gene therapies, agriculture, smart cities, and smart homes are just a very few of the categorical examples where IoT is strongly established.

# Definition:-

# IoT is network of interconnected computing devices which are embedded in everyday objects, enabling them to send and receive data.

# Over 9 billion ‘Things’ (physical objects) are currently connected to the Internet, as of now. In the near future, this number is expected to rise to a whopping 20 billion.

# 1.2 HISTORY OF IoT

# In 1991, Mark Weiser has described the vision of the future Internet under the name of “Ubiquitous Computing”. Through this vision he was focused on how to turn on the smart liveable environment in the presence of mobile phone technology this provide a powerful multimedia system [6]. Kevin Ashton is a one of the pioneers talk about IoT [2]. According to Atzori Alera et al [7], classified IoT to three paradigms namely,

# i) Internet oriented (Middleware),

# ii) Things oriented (Sensors), and

# iii) Semantic oriented (Knowledge).

# In 1999 Neil Gershenfeld was speaking about similar things from the Massachusetts Institute of Technology, MIT Media Lab in his book “When Things Start to think”. In 1999 Auto- ID labs and MIT sought to develop Electronic Product Code EPC, and use RFID to identify things on the network. In 2003-2004 the emergence of projects serving IoT idea such as Cooltown, Internet0, and the Disappearing Computer initiative, also IoT start to appear in book titles for the first time. RFID is deployed was published on a massive scale by the US Department of Defence. In 2005 IoT entered a new level when published its first report by International Telecommunication Union (ITU). In 2008 a group of companies such as Cisco, Intel, SAP and over 50 other members of companies met to create IPSO Alliance, to promote the use of Internet protocol (IP) and to activate IoT concept. In 2008-2009 IoT was "Born" by Cisco Internet Business Solutions Group (IBSG) [8]. From the previous perspectives can be defined IoT as a set of smart things/objects such as home devices, mobile, laptop, etc., addressed by a unique addressing scheme and connected to the Internet through a unified framework this framework may be cloud computing. Fig 1 depicts IoT technology

# 1.3 Main components used in IoT:

# Low-power embedded systems: Less battery consumption, high performance are the inverse factors that play a significant role during the design of electronic systems.

# Sensors: Sensors are the major part of any IoT applications. It is a physical device that measures and detect certain physical quantity and convert it into   signal which can be provide as an input to processing or control unit for analysis purpose.

# 1.3.1 Different types of Sensors:

# Temperature Sensors

# Image Sensors

# Gyro Sensors

# Obstacle Sensors

# RF Sensor

# IR Sensor

# MQ-02/05 Gas Sensor

# LDR Sensor

# Ultrasonic Distance Sensor

# 1.4 Control Units:

# It is a unit of small computer on a single integrated circuit containing microprocessor or processing core, memory and programmable input/output devices/peripherals. It is responsible for major processing work of IoT devices and all logical operations are carried out here.

# 1.4.1 Cloud computing:

# Data collected through IoT devices is massive and this data has to be stored on a reliable storage server. This is where cloud computing comes into play. The data is processed and learned, giving more room for us to discover where things like electrical faults/errors are within the system.

# 1.4.2 Availability of big data:

# We know that IoT relies heavily on sensors, especially in real-time. As these electronic devices spread throughout every field, their usage is going to trigger a massive flux of big data.

# 1.4.3 Networking connection:

# In order to communicate, internet connectivity is a must where each physical object is represented by an IP address. However, there are only a limited number of addresses available according to the IP naming. Due to the growing number of devices, this naming system will not be feasible anymore. Therefore, researchers are looking for another alternative naming system to represent each physical object.

# There are two ways of building IoT:

# Form a separate internetwork including only physical objects.

# Make the Internet ever more expansive, but this requires hard-core technologies such as rigorous cloud computing and rapid big data storage (expensive).

# In the near future, IoT will become broader and more complex in terms of scope. It will change the world in terms of “anytime, anyplace, anything in connectivity.”

# 1.5 IoT Enablers:

# RFIDs: uses radio waves in order to electronically track the tags attached to each physical object.

# Sensors: devices that are able to detect changes in an environment (ex: motion detectors).

# Nanotechnology: as the name suggests, these are extremely small devices with dimensions usually less than a hundred nanometres.

# Smart networks: (ex: mesh topology).

# 1.6 Working with IoT Devices:

# Collect and Transmit Data*:*For this purpose sensors are widely used they are used as per requirements in different application areas.

# Actuate device based on triggers produced by sensors or processing devices*:*If certain condition is satisfied or according to user’s requirements if certain trigger is activated then which action to performed that is shown by Actuator devices.

# Receive Information*:*From network devices user or device can take certain information also for their analysis and processing purposes*.*

# *Communication Assistance:*Communication assistance is the phenomena of communication between 2 networks or communication between 2 or more IoT devices of same or different Networks. This can be achieved by different communication protocols like: MQTT, Constrained Application Protocol, ZigBee, FTP, HTTP etc.

# https://media.geeksforgeeks.org/wp-content/uploads/20220528223543/iotworking.PNG

# Figure 1.1

# Characteristics of IoT:

# Massively scalable and efficient

# IP-based addressing will no longer be suitable in the upcoming future.

# An abundance of physical objects is present that do not use IP, so IoT is made possible.

# Devices typically consume less power. When not in use, they should be automatically programmed to sleep.

# A device that is connected to another device right now may not be connected in another instant of time.

# Intermittent connectivity – IoT devices aren’t always connected. In order to save bandwidth and battery consumption, devices will be powered off periodically when not in use. Otherwise, connections might turn unreliable and thus prove to be inefficient.

# Desired Quality of any IoT Application:

# Interconnectivity:-

# It is the basic first requirement in any IoT infrastructure.

# Connectivity should be guaranteed from any devices on any

# network then only devices in a network can communicate with each other.

# Heterogeneity:-

# There can be diversity in IoT enabled devices like different hardware and

# software configuration or different network topologies or connections but they should connect and interact with each other despite of so much heterogeneity.

# Dynamic in nature

# IoT devices should dynamically adapt themselves to the

# Changing surroundings like different situation and different prefaces.

# Self-adapting and self-configuring technology

# For example surveillance camera. It should be flexible to work

# in different weather conditions and different light situations

# (morning, afternoon, or night).

# Intelligence

# Just data collection is not enough in IoT, extraction of knowledge from the generated data is very important. For example, sensors generate data, but that data will only be useful if it is interpreted properly. So intelligence is one of the key characteristics in IoT. Because data interpretation is the major part in any IoT application because without data processing we can’t make any insights from data. Hence big data is also one of the most enabling technology in IoT field.

# Scalability

# The number of elements (devices) connected to IoT zone is increasing day by day. Therefore, an IoT setup should be capable of handling the expansion. It can be either expand capability in terms of processing power, Storage, etc. as vertical scaling or horizontal scaling by multiplying with easy cloning

# Identity

# Each IoT device has a unique identity (e.g., an IP address). This identity is helpful in communication, tracking and to know status of the things. If there is no identification then it will directly effect security and safety of any system because without discrimination we can’t identify with whom one network is connected or with whom we have to communicate. So there should be clear and appropriate discrimination technology available between IoT networks and devices.

# Safety

# Sensitive personal details of a user might be compromised when the devices are connected to the Internet. So data security is a major challenge. This could cause a loss to the user. Equipment in the huge IoT network may also be at risk. Therefore, equipment safety is also critical.

# Architecture

# It should be hybrid, supporting different manufacturer’s products to function in the IoT network*.*

# As a quick note, IoT incorporates trillions of sensors, billions of smart systems, and millions of applications.

# Application Domains: IoT is currently found in four different popular domains:

# 1) Manufacturing/Industrial business - 40.2%

# 2) Healthcare - 30.3%

# 3) Security - 7.7%

# 4) Retail - 8.3%

# 1.7 Modern Applications:

# Smart Grids and energy saving

# Smart cities

# Smart homes/Home automation

# Healthcare

# Earthquake detection

# Radiation detection/hazardous gas detection

# Smartphone detection

# Water flow monitoring

# Traffic monitoring

# Wearable’s

# Smart door lock protection system

# Robots and Drones

# Healthcare and Hospitals, Telemedicine applications

# Security

# Biochip Transponders (For animals in farms)

# Heart monitoring implants (Example Pacemaker, ECG real time tracking)

**CHAPTER 2**

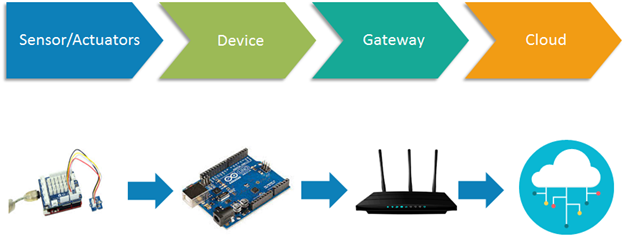
**Architecture and Domains**

# 2.1 IoT Architecture:-

There is not such a unique or standard consensus on the Internet of Things (IoT) architecture which is universally defined. The IoT architecture differs from their functional area and their solutions. However, the IoT architecture technology mainly consists of four major components:

Components of IoT Architecture

* Sensors/Devices
* Gateways and Networks
* Cloud/Management Service Layer
* Application Layer



**Figure 2.1**

**2.1.1 Stages of IoT Solutions Architecture:-**

There are several layers of IoT built upon the capability and performance of IoT elements that provides the optimal solution to the business enterprises and end-users. The IoT architecture is a fundamental way to design the various elements of IoT, so that it can deliver services over the networks and serve the needs for the future.

Following are the primary stages (layers) of IoT that provides the solution for IoT architecture.

1. **Sensors/Actuators:** Sensors or Actuators are the devices that are able to emit, accept and process data over the network. These sensors or actuators may be connected either through wired or wireless. This contains GPS, Electrochemical, Gyroscope, RFID, etc. Most of the sensors need connectivity through sensors gateways. The connection of sensors or actuators can be through a Local Area Network (LAN) or Personal Area Network.
2. **Gateways and Data Acquisition:** As the large numbers of data are produced by this sensors and actuators need the high-speed Gateways and Networks to transfer the data. This network can be of type Local Area Network (LAN such as Wi-Fi, Ethernet, etc.), Wide Area Network (WAN such as GSM, 5G, etc.).
3. **Edge IT:** Edge in the IoT Architecture is the hardware and software gateways that analyse and pre-process the data before transferring it to the cloud. If the data read from the sensors and gateways are not changed from its previous reading value then it does not transfer over the cloud, this saves the data used.
4. **Data center/ Cloud:** The Data Center or Cloud comes under the Management Services which process the information through analytics, management of device and security controls. Beside this security controls and device management the cloud transfer the data to the end users application such as Retail, Healthcare, Emergency, Environment, and Energy, etc.

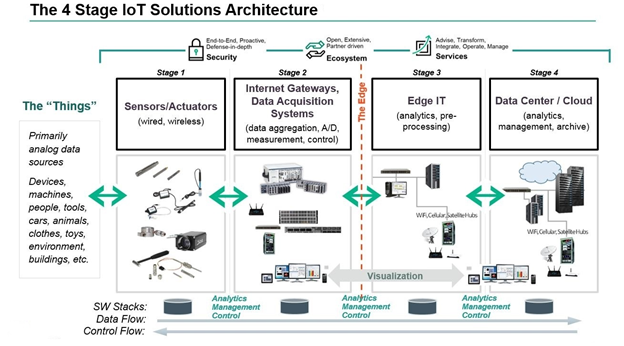


Figure 2.2

# 2.2 Domains in IoT:-

# 2.2.1. IoT in Smart Home and Smart City Application:-

Implementing IoT system in home and city leads them to become as smart home and smart city. Smart home or smart city make life quite easier and smarter.

A **smart home system** can be something that makes our life quite easy. Starting from energy management where the power controls system in the AC appliances where we use the thermostat, all this is managed to cut down the power consumption that's taking place. A door management system, security management system, water management system are the part of this as well. Still, these are vital things that stand out in the smart home system. The limitation of IoT in smart home application stops where our imagination stops. Anything that we wish to automate or want to make our life easier can be a part of smart home, a smartphone system as well.

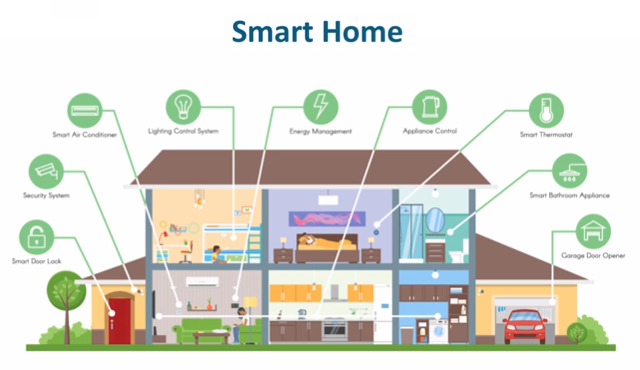


Figure 2.3

Now, a smart home usually is going to be a base of a **smart city**. The smart city is an evolution of a smart home. Here, it is not just the sensors of a single home that is connected, here its correlation or a network or a connection between various organizations, various domains as well as multiple segments of that city as a whole. In the smart city, the life of every single dependent becomes more comfortable and in tune really help to develop that city to greater extends as such. Now, the key factor for a smart city is government support as well, and if the governments are willing to take this step, then we hope we would see a smart city completely build on the Internet of Things.

# IoT Smart Home and Smart City Application

Figure 2.4

**2.2.2** **IoT in Security Camera & Door Unlock System:-**

The Security Camera and a Door unlock system is something that is quite interesting IoT application. The phenomena of its working process is briefly mentioned here.

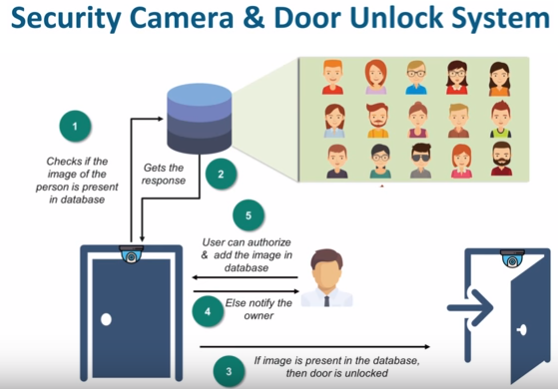


Figure 2.5

**How does this system work?**

Here, we place a camera on the top foot of the door which in turn clicks the photo of a person who comes into frame. Now, this photo is sent to an analytical system which in turn compares this with all the photos it possesses to identify whether to let the user open the door or not.

Now, if it does not find the photo of that person then it can notify the concern that a person is trying to access this door would you like to authorize this person? or would you like to deny the access to this person?

Usually, the Security Camera and a Door unlock system is used in the areas where you have highly sensitive information stored. Another usage of the security camera and door unlock system can be at our homes when we want to identify who comes to our home when we are not there and either decide to give them access to our home or not.

**2.3** **TECHNOLOGIES INVOLVED IN IOT DEVELOPMENT:**

**2.3.1 INTERNET/WEB AND NETWORKING BASICS OSI MODEL:-**

• Networking technologies enable IoT devices to communicate with other devices, applications, and services running in the cloud.

• The internet relies on standardized protocols to ensure communication between heterogeneous devices is secure and reliable. • Standard protocols specify rules and formats that devices use to establish and manage networks and transmit data across those networks.

• Networks are built as a “stack” of technologies. A technology such as Bluetooth LE is at the bottom of the stack.

• While others such as such as IPv6 technologies (which is responsible for the logical device addressing and routing of network traffic) are further up the stack. Technologies at the top of the stack are used by the applications that are running on top of those layers, such as message queuing technologies.

• This article describes widely adopted technologies and standards for IoT networking. It also provides guidance for choosing one network protocol over another. It then discusses key considerations and challenges related to networking within IoT: range, bandwidth, power usage, intermittent connectivity, interoperability, and security.

**2.3.2 NETWORKING STANDARDS AND TECHNOLOGIES**:-

• The Open Systems Interconnection (OSI) model is an ISO-standard abstract model is a stack of seven protocol layers.

• From the top down, they are: application, presentation, session, transport, network, data link and physical. TCP/IP, or the Internet Protocol suite, underpins the internet, and it provides a simplified concrete implementation of these layers in the OSI model. Figure 1. OSI and TCP/IP networking models The TCP/IP model includes only four layers, merging some of the OSI model layers:

• **2.3.3** **Network Access & Physical Layer: -** This TCP/IP Layer subsumes both OSI layers 1 and 2. The physical (PHY) layer (Layer 1 of OSI) governs how each device is physically connected to the network with hardware, for example with an optic cable, wires, or radio in the case of wireless network like Wi-Fi IEEE 802.11 a/b/g/n). At the link layer (Layer 2 of OSI), devices are identified by a MAC address, and protocols at this level are concerned with physical addressing, such as how switches deliver frames to devices on the network.

• **Internet Layer: -** This layer maps to the OSI Layer 3 (network layer). OSI Layer 3 relates to logical addressing. Protocols at this layer define how routers deliver packets of data between source and destination hosts identified by IP addresses. IPv6 is commonly adopted for IoT device addressing.

• **Transport Layer**:-

The transport layer (Layer 4 in OSI) focuses on end-to-end communication and provides features such as reliability, congestion avoidance, and guaranteeing that packets will be delivered in the same order that they were sent. UDP (User Datagram protocol) is often adopted for IoT transport for performance reasons.

• **Application Layer**: - The application layer (Layers 5, 6, and 7 in OSI) covers application-level messaging. HTTP/S is an example of an application layer protocol that is widely adopted across the internet. Although the TCP/IP and OSI models provide you with useful abstractions for discussing networking protocols and specific technologies that implement each protocol, some protocols don’t fit neatly into these layered models and are impractical. For example, the Transport Layer Security (TLS) protocol that implements encryption to ensure privacy and data integrity of network traffic can be considered to operate across OSI layers 4, 5, and 6.

**2.3.4 NETWORK** **ACCESS AND PHYSICAL LAYER IOT NETWORK** **TECHNOLOGIES:-**

IoT network technologies to be aware of toward the bottom of the protocol stack include cellular, Wi-Fi, and Ethernet, as well as more specialized solutions such as LPWAN, Bluetooth Low Energy (BLE), ZigBee, NFC, and RFID. NB-IoT is becoming the standard for LPWAN networks, according to Gartner. This IoT for All article tells more about NB-IoT. The following are network technologies with brief descriptions of each:

• **LPWAN (Low Power Wide Area Network):-**

It is a category of technologies designed for low power, long-range wireless communication. They are ideal for large-scale deployments of low-power IoT devices such as wireless sensors. LPWAN technologies include LoRa (LongRange physical layer protocol), Haystack, SigFox, LTE-M, and NB-IoT (Narrow-Band IoT).

• **Cellular The LPWAN NB-IoT and LTE-M standards**:-

NB-IoT is the newest of these standards and is focused on long-range communication between large numbers of primarily indoor devices. LTE-M and NB-IoT were developed specifically for IoT, however existing cellular technologies are also frequently adopted for long-range wireless communication. While this has included 2G (GSM) in legacy devices (and currently being phased out), CDMA (also being retired or phased out), it also includes 3G, which is rapidly being phased out with several network providers retiring all 3G devices. 4G is still active and will be until 5G becomes fully available and implemented.

• **Bluetooth Low Energy (BLE):-**

BLE is a low-power version of the popular Bluetooth 2.4 GHz wireless communication protocol. It is designed for short-range (no more than 100 meters) communication, typically in a star configuration, with a single primary device that controls several secondary devices. Bluetooth operates across both layers 1 (PHY) and 2 (MAC) of the OSI model. BLE is best suited to devices that transmit low volumes of data in bursts. Devices are designed to sleep and save power when they are not transmitting data. Personal IoT devices such as wearable health and fitness trackers, often use BLE.

• **ZigBee:-**

ZigBee operates on 2.4GHz wireless communication spectrum. It has a longer range than BLE by up to 100 meters. It also has a slightly lower data rate (250 kbps maximum compared to 270 kbps for BLE) than BLE. ZigBee is a mesh network protocol. Unlike BLE, not all devices can sleep between bursts. Much depends on their position in the mesh and whether they need to act as routers or controllers within the mesh. ZigBee was designed for building and home automation applications. Another closely related technology to ZigBee is Z-Wave, which is also based on IEEE 802.15.4. Z-Wave was designed for home automation. It has been proprietary technology, but was recently released as a public domain specification.

• **NFC**:-

The near field communication (NFC) protocol is used for very small range communication (up to 4 cm), such as holding an NFC card or tag next to a reader. NFC is often used for payment systems, but also useful for check-in systems and smart labels in asset tracking.

• **RFID**: -

RFID stands for Radio Frequency Identification. RFID tags store identifiers and data. The tags are attached to devices and read by an RFID reader. The typical range of RFID is less than a meter. RFID tags can be active, passive, or assisted passive. Passive tags are ideal for devices without batteries, as the ID is passively read by the reader. Active tags periodically broadcast their ID, while assisted passive tags become active when RFID reader is present. Dash7 is a communication protocol that uses active RFID that is designed to be used within Industrial IoT applications for secure long-range communication. Similar to NFC, a typical use case for RFID is tracking inventory items within retail and industrial IoT applications. • **Wi-Fi**:

Wi-Fi is standard wireless networking based on IEEE 802.11a/b/g/n specifications. 802.11n offers the highest data throughput, but at the cost of high-power consumption, so IoT devices might only use 802.11b or g for power conservation reasons. Although wifi is adopted within many prototype and current generation IoT devices, as longer-range and lower-power solutions become more widely available, it is likely that wifi will be superseded by lower-power alternatives.

• **Ethernet:-**

Widely deployed for wired connectivity within local area networks, Ethernet implements the IEEE 802.3 standard. Not all IoT devices need to be stationery wireless. For example, sensor units installed within a building automation system can use wired networking technologies like Ethernet. Power line communication (PLC), an alternative hard-wired solution, uses existing electrical wiring instead of dedicated network cables.

* **INTERNET LAYER IOT NETWORK TECHNOLOGIES: -** Internet layer technologies (OSI Layer 3) identify and route packets of data. Technologies commonly adopted for IoT are related to this layer, and include IPv6, 6LoWPAN, and RPL.

•  **IPv6**:-

At the Internet layer, devices are identified by IP addresses. IPv6 is typically used for IoT applications over legacy IPv4 addressing. IPv4 is limited to 32-bit addresses, which only provide around 4.3 billion addresses in total, which is less than the current number of IoT devices that are connected, while IPv6 uses 128 bits, and so provides 2 128 addresses (around 3.4 × 10 38 or 340 billion billion billion billion) addresses. In practice, not all IoT devices need public addresses. Of the tens of billions of devices expected to connect via the IoT over the next few years, many will be deployed in private networks that use private address ranges and only communicate out to other devices or services on external networks by using gateways.

• **6LoWPAN:-**

The IPv6 Low Power Wireless Personal Area Network (6LoWPAN) standard allows IPv6 to be used over 802.15.4 wireless networks. 6LoWPAN is often used for wireless sensor networks, and the Thread protocol for home automation devices also runs over 6LoWPAN.

• **RPL**:-

The Internet Layer also covers routing. IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) is designed for routing IPv6 traffic over low-power networks like those networks implemented over 6LoWPAN. RPL (pronounced “ripple”) is designed for routing packets within constrained networks such as wireless sensor networks, where not all devices are reachable at all times and there are high or unpredictable amounts of packet loss. RPL can compute the optimal path by building up a graph of the nodes in the network based on dynamic metrics and constraints like minimizing energy consumption or latency.

**2.3.5 APPLICATION LAYER IOT NETWORK TECHNOLOGIES:-**

HTTP and HTTPS are ubiquitous across internet applications, which is true also within IoT, with Restful HTTP and HTTPS interfaces widely deployed. CoAP (Constrained Application Protocol) is like a lightweight HTTP that is often used in combination with 6LoWPAN over UDP. Messaging protocols like MQTT, AMQP, and XMPP are also frequently used within IoT applications:

• **MQTT**:-

Message Queue Telemetry Transport (MQTT) is a publish/subscribe-based messaging protocol that was designed for use in low bandwidth situations, particularly for sensors and mobile devices on unreliable networks.

• **AMQP**:-

Advanced Message Queuing Protocol (AMQP) is an open standard messaging protocol that is used for message-oriented middleware. Most notably, AMQP is implemented by RabbitMQ.

• **XMPP:-**

The Extensible Messaging and Presence Protocol (XMPP) was originally designed for real-time human-to-human communication including instant messaging. This protocol has been adapted for machine-to-machine (M2M) communication to implement lightweight middleware and for routing XML data. XMPP is primarily used with smart appliances. Your choice of technologies at this layer will depend on the specific application requirements of your IoT project. For example, for a budget home automation system that involves several sensors, MQTT would be a good choice as it is great for implementing messaging on devices without much storage or processing power because the protocol is simple and lightweight to implement.

**2.4 IOT NETWORKING CONSIDERATIONS AND CHALLENGES**:-

When you consider which networking technologies to adopt within your IoT application, be mindful of the following constraints:

• Range

• Bandwidth

• Power usage

• Intermittent connectivity

• Interoperability

• Security Range

**Bandwidth**: -

Bandwidth is the amount of data that can be transmitted per unit of time. It limits the rate at which data can be collected from IoT devices and transmitted upstream. Bandwidth is affected by many factors, which include:

• The volume of data each device gathers and transmits

• The number of devices deployed

• Whether data is being sent as a constant stream or in intermittent bursts, and if any peak periods are notable

The packet size of the networking protocol should match up with the volume of data typically transmitted. It is inefficient to send packets padded with empty data. In contrast, there are overheads in splitting larger chunks of data up across too many small packets. Data transmission rates are not always symmetrical (that is, upload rates might be slower than download rates). So, if there is two-way communication between devices, data transmission needs to be factored in. Wireless and cellular networks are traditionally low bandwidth, so consider whether a wireless technology is the right choice for high-volume applications. Consider whether all raw data must be transmitted. A possible solution is to capture less data by sampling less frequently. Thus, you’ll capture fewer variables and may filter data from the device to drop insignificant data. If you aggregate the data before you transmit it, you reduce the volume of data transmitted. But this process affects flexibility and granularity in the upstream analysis. Aggregation and bursting are not always suitable for time-sensitive or latency-sensitive data. All of these techniques increase the data processing and storage requirements for the IoT device.

**Power usage:-**

Transmitting data from a device consumes power. Transmitting data over long ranges requires more power than over a short range. You must consider the power source – such as a battery, solar cell, or capacitor – of a device and its total lifecycle. A long and enduring lifecycle will not only provide greater reliability but reduce operating cost. Steps may be taken to help achieve longer power supply lifecycles. For example, to prolong the battery life, you can put the device into sleep mode whenever it is idle. Another best practice is to model the energy consumption of the device under different loads and different network conditions to ensure that the device’s power supply and storage capacity matches with the power that is required to transmit the necessary data by using the networking technologies that you adopted. Intermittent connectivity IoT devices aren’t always connected. In some cases, devices are designed to connect periodically. However, sometimes an unreliable network might cause devices to drop off due to connectivity issues. Sometimes quality of service issues, such as dealing with interference or channel contention on a wireless network using a shared spectrum. Designs should incorporate intermittent connectivity and seek any available solutions to provide uninterrupted service, should that be a critical factor for IoT landscape design. Interoperability Devices work with other devices, equipment, systems, and technology; they are interoperable. With so many different devices connecting to the IoT, interoperability can be a challenge. Adopting standard protocols has been a traditional approach for maintaining interoperability on the Internet. Standards are agreed upon by industry participants and avoid multiple different designs and directions. With proper standards, and participants who agree to them, incompatibility issues, hence interoperability issues may be avoided. However, for the IoT, standardization processes sometimes struggle to keep up with innovation and change. They are written and released based on upcoming versions of standards that are still subject to change.

**Security**:-Security is a priority. Selection of networking technologies that implement end-to-end security, including authentication, encryption, and open port protection is crucial. IEEE 802.15.4 includes a security model that provides security features that include access control, message integrity, message confidentiality, and replay protection, which are implemented by technologies based on this standard such as ZigBee.

**2.5 Consider the following factors in shaping a secure and safe IoT network:**

• **Authentication**:-

Adopt secure protocols to support authentication for devices, gateways, users, services, and applications. Consider using adopting the X.509 standard for device authentication.

• **Encryption**:-

If you are using Wi-Fi, use Wireless Protected Access 2 (WPA2) for wireless network encryption. You may also adopt a Private Pre-Shared Key (PPSK) approach. To ensure privacy and data integrity for communication between applications, be sure to adopt TLS or Datagram Transport-Layer Security (DTLS), which is based on TLS, but adapted for unreliable connections that run over UDP. TLS encrypts application data and ensures its integrity.

• **Port protection**:-

Port protection ensures that only the ports required for communication with the gateway or upstream applications or services remain open to external connections. All other ports should be disabled or protected by firewalls. Device ports might be exposed when exploiting Universal Plug and Play (UPnP) vulnerabilities. Thus, UPnP should be disabled on the router.

The IoT World Forum (IoTWF) Standardized Architecture In 2014 the IoTWF architectural committee (led by Cisco, IBM, Rockwell Automation, and others)published a seven-layer IoT architectural reference model. While various IoT reference models exist, the one put forth by the IoT World Forum offers a clean, simplified perspective on IoT and includes edge computing, data storage, and access. It provides a succinct way of visualizing IoT from a technical perspective. Each of the seven layers is broken down into specific functions, and security encompasses the entire model.

**CHAPTER 3**

**IoT Devices**

**3.1 What Are IoT Devices?**

* + - IoT devices are hardware devices, such as sensors, gadgets, appliances and other machines that collect and exchange data over the Internet. They are programmed for certain applications and can be embedded into other IoT devices. For example, an IoT device in your car can identify the traffic ahead and send out a message automatically to the person you are about to meet of your impending delay.

.

**3.1.1 Examples of IoT Devices:-**

**Home Security:-**

* + - The key driver behind smart and secure homes is IoT. A variety of sensors, lights, alarms and cameras (all of which can be controlled from a smartphone) are connected via IoT to provide 24x7 security.



Figure 3.1

Source: - [https://psiborg.in/how-iot-is-used-for-home-security-and-why-it- is-required/](https://psiborg.in/how-iot-is-used-for-home-security-and-why-it-%20is-required/)

**Activity Trackers:-**

* + - Smart home security cameras provide alerts and peace of mind. Activity trackers are sensor devices that can monitor and transmit key health indicators in real-time. You can track and manage your blood pressure, appetite, physical movement and oxygen levels.



Figure 3.2

Source: - <https://www.pinterest.com/pin/463870830369900075/>

**Industrial Security and Safety:-**

* + - IoT-enabled detection systems, sensors and cameras can be placed in restricted areas to detect trespassers. They can also identify pressure buildups and small leaks of hazardous chemicals and fix them before they become serious problems.

**Augmented Reality Glasses:-**

* + - Augmented Reality (AR) glasses are wearable computer-enabled glasses that help you get extra information such as 3D animations and videos to the user’s real-world scenes. The information is presented within the lenses of the glasses and can help users access Internet applications.



Figure 3.3

Source: - <https://www.pinterest.com/pin/463870830369900075/>

**Motion Detection:-**

* + - Motion sensors can detect vibrations in buildings, bridges, dams and other large-scale structures. These devices can identify anomalies and disturbances in the structures that could lead to catastrophic failures. They can also be used in areas susceptible to floods, landslides, and earthquakes.

# How do IoT devices work?

# IoT devices vary in terms of functionality, but IoT devices have some similarities in how they work. First, IoT devices are physical objects designed to interact with the real world in some way. The device might be a sensor on an assembly line or an intelligent security camera. In either case, the device is sensing what's happening in the physical world.

# The device itself includes an integrated CPU, network adapter and firmware, which is usually built on an open source platform. In most cases, IoT devices connect to a Dynamic Host Configuration Protocol server and acquire an IP address that the device can use to function on the network. Some IoT devices are directly accessible over the public internet, but most are designed to operate exclusively on private networks.

# Although not an absolute requirement, many IoT devices are configured and managed through a software application. Some devices, however, have integrated web servers, thus eliminating the need for an external application.

# Once an IoT device has been configured and begins to operate, most of its traffic is outbound. A security camera, for example, streams video data. Likewise, an industrial sensor streams sensor data. Some IoT devices such as smart lights, however, do accept inputs

.

**3.3 What is IoT device management?**

Several challenges can hinder the successful deployment of an IoT system and its connected devices, including security, interoperability, power/processing capabilities, scalability and availability. Many of these can be addressed with IoT device management either by adopting standard protocols or using services offered by a vendor.

Device management helps companies integrate, organize, monitor and remotely manage internet-enabled devices at scale, offering features critical to maintaining the health, connectivity and security of the IoT devices along their entire lifecycles. Such features include:

* Device registration and activation
* Device authentication/authorization
* Device configuration
* Device provisioning
* Device monitoring and diagnostics
* Device troubleshooting
* Device firmware updates

Available standardized device management protocols include the Open Mobile Alliance's Device Management and Lightweight Machine-to-Machine.

IoT device management services and software are also available from vendors, including Amazon, Microsoft, Google, IBM, GE and many others.

**3.4 IoT device connectivity and networking**

The networking, communication and connectivity protocols used with internet-enabled devices largely depend on the specific IoT application deployed. Just as there are many different IoT applications, there are many different connectivity and communication options.

Communication protocols include CoAP, DTLS, MQTT, DDS and AMQP. Wireless protocols include IPv6, LPWAN, Zigbee, Bluetooth Low Energy, Z-Wave, RFID and NFC. Cellular, satellite, Wi-Fi and Ethernet can also be used.

Each option has its trade-offs in terms of power consumption, range and bandwidth, all of which must be considered when choosing connected devices and protocols for a particular IoT application.

In most cases, IoT devices connect to an IoT gateway or another edge device where data can either be analysed locally or sent to the cloud for analysis. Some devices have integrated data processing capabilities that minimized the amount of data that must be sent to the cloud or to the data center. This type of processing often uses machine learning capabilities that are integrated into the device, and is becoming increasingly popular as IoT devices create more and more data.

**3.5 What security risks do IoT devices pose?**

The interconnection of traditionally dumb devices raises several questions in relation to security and privacy. As is often the case, IoT technology has moved more quickly than the mechanisms available to safeguard devices and their users.

Researchers have already demonstrated remote hacks on pacemakers and cars. In October 2016, a large distributed denial-of-service attack dubbed Mirai affected DNS servers on the east coast of the United States, disrupting services worldwide -- an issue traced back to hackers infiltrating networks through IoT devices, including wireless routers and connected cameras. Similarly, in 2015 a team of researchers proved that it was possible to take control of a Jeep by exploiting a cellular network and the vehicle's Controller Area Network bus.

However, safeguarding IoT devices and the networks they connect to can be challenging due to the variety of devices and vendors, as well as the difficulty of adding security to resource-constrained devices. In the case of the Mirai botnet, the problem was traced back to the use of default passwords on the hacked devices. Strong passwords, authentication/authorization and identity management, network segmentation, encryption and cryptography are all suggested IoT security measures.

Concerned by the dangers posed by the rapidly growing IoT attack surface, the FBI released the public service announcement FBI Alert Number I-091015-PSA in September 2015, which is a document outlining the risks of IoT devices, as well as protections and defense recommendations.

In August 2017, the U.S. Senate introduced the IoT Cybersecurity Improvement Act, a bill addressing security issues associated with IoT devices. A subsequent bill called the Internet of Things Cybersecurity Improvement Act of 2020 sought to further improve IoT related cybersecurity by requiring NIST to develop and publish standards and guidelines on the use and management of IoT devices. Although these standards are intended for use by federal agencies, they will almost certainly be adopted by the private sector as well.

**3.6 IoT device trends and anticipated growth:-**

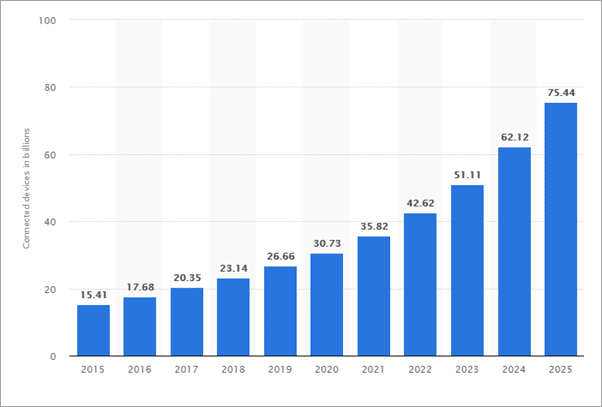
The estimations for future growth of IoT devices have been fast and furious. At the high end of the scale, Intel projected that internet-enabled device penetration would grow from 2 billion in 2006 to 200 billion by 2020, which equates to nearly 26 smart devices for each human on Earth. A little more conservative, IHS Markit said the number of connected devices will be 75.4 billion in 2025 and 125 billion by 2030.

Other companies have tempered their numbers, taking smartphones, tablets and computers out of the equation. Gartner estimated 20.8 billion connected things would be in use by 2020, with IDC coming in at 28.1 billion and BI Intelligence at 24 billion.

Gartner estimated total spending on IoT devices and services at nearly $2 trillion in 2017, with IDC projecting spending to reach $772.5 billion in 2018, 14.6% more than the $674 billion it estimated to be spent in 2017, with it hitting $1 trillion in 2020 and $1.1 trillion in 2021.

The key to making effective use of IoT devices is to make sure that you start your IoT strategy on the right foot and that you understand how the edge and IoT are intertwined with one another. Regardless of whether you have IoT devices already in use or are considering adopting IoT devices in your organization, make sure you're prepared to handle the unique security challenges presented by IoT devices.

**Growth Graph:-**



**Figure 3.4**

**Source: -** <https://www.softwaretestinghelp.com/iot-devices/>

**CHAPTER4**

**Sensors**

# 4.1 What are IoT Sensors?

# IoT sensors are pieces of hardware that detect changes in an environment

# and collect data.

# They’re the pieces of an IoT ecosystem that bridge the digital world to the physical world.

# IoT sensors may detect things like temperature, pressure, and motion,

# and if they are connected to a network, they share data with the network.

# Types of IoT Sensors:-

# There are many different types of sensors, and they come in different

# shapes and sizes.

# Here are 14 of the most common types and uses of sensors.

# 1. Temperature Sensors

# Temperature sensors measure the amount of heat generated from an area or an object. They detect a temperature change and convert the findings to data. Temperature sensors are used in various industries, including manufacturing, healthcare, and agriculture. Some examples are thermistors, thermocouples, and resistor temperature detectors (RTD).

# 

# Figure 4.1

# Source:- <https://www.sunrom.com/p/temperature-sensor-module-ntc>

# 2. Proximity Sensors

# Proximity sensors detect the presence or absence of objects near the sensor without physical contact. They often emit a beam of radiation like infrared or an electromagnetic field. They can be used for process monitoring and control, object counting, assembly lines, and determining available space. Proximity sensors are common in retail settings, industrial complexes, and parking lots. Some examples are photoelectric, magnetic, capacitive, inductive, and ultrasonic.

# 3. Pressure Sensors

# These sensors detect changes in a gas or liquid. When the pressure range is beyond a set threshold, pressure sensors alert to the problem. They are used for leak testing, water systems, vehicles, and aircraft. For example, the BMP180 is a digital pressure sensor found in cell phones and GPS navigation devices. And some vehicles use a tire pressure monitoring system (TPMS) to alert when tire pressure is low and potentially unsafe.

# 4. Water Quality Sensors

# As you’d expect, water quality sensors monitor the quality of water. They are often used in water distribution systems, but they function in a variety of industries. There are different kinds of water sensors, including residual chlorine sensors, turbidity sensors, pH sensors, and total organic carbon sensors.

# 5. Chemical and Gas Sensors

# These sensors monitor air quality for the presence of toxic or hazardous gas. They often use semiconductor, electrochemical, or photo-ionization technologies for detection. They are typically used in industrial and manufacturing settings, though they are also found in carbon dioxide detectors.

# 6. Infrared Sensors

# Some sensors either detect or emit infrared radiation to sense characteristics and changes in the surrounding area. They’re useful for measuring heat emissions from an object. Infrared sensors are used in remote controls, healthcare settings, and even by art historians authenticating artwork.

# 7. Smoke Sensors

# Most people are familiar with smoke detectors, as they have protected our homes and businesses for a long time. However, with improvements based on IoT, smoke detectors are now more user-friendly, convenient, and wire-free.

# 8. Motion Sensors

# Motion sensors detect physical movement in an area. Of course, these sensors play a significant role in the security industry, but they are used in nearly every industry. Applications include automated sinks and toilet flushers, automatic door controls, energy management systems, and automated parking systems. Standard motion sensors include ultrasonic, microwave, and passive infrared (PIR).

# 9. Level Sensors

# Level sensors detect the level of various substances, including powder, granular material, and liquids. Industries that use them include water treatment, food and beverage manufacturing, oil manufacturing, and waste management. They can detect the level of liquid in a container and can even determine the amount of waste in a dumpster.

# 10. Image Sensors

# These sensors convert optical images into signals and are generally used to display or store files electronically. They are found in radar and sonar, biometric devices, night vision equipment, medical imaging, digital cameras, and even some cars. Charge-coupled devices (CCD) and complementary metal-oxide semiconductors (CMOS) are most commonly used.

# 11. Humidity Sensors

# These sensors measure the amount of water vapour in the air. Typical uses include heating and air conditioning systems (HVAC) and weather monitoring and prediction. When humidity must be tightly controlled, such as in museums, hospitals, and greenhouses, humidity sensors assist the process.

# 12. Accelerometer Sensors

# Accelerometer sensors detect the orientation of an object and the rate of change, including tap, shake, tilt, and positioning. They are used in many industries for smart pedometers, anti-theft protection, and monitoring auto fleets. Some types are capacitive accelerometers and hall-effect accelerometers.

# 13. Gyroscope Sensors

# A gyroscope sensor measures the angular rate or velocity, or the speed of rotation around an axis. They are generally used for navigation in the auto industry for navigation and anti-skid systems as well as in video games and drones. Some examples include optical gyroscopes, rotary gyroscopes, and vibrating structure gyroscopes.

# 14. Optical Sensors

# Optical sensors measure light and convert it into electrical signals. Many industries make use of optical sensors, including auto, energy, healthcare, and aerospace. Sensors include fibre optics, photo detector, and pyrometer.

# IoT Sensors Help Drive the Value of IoT:-

# Sensors can monitor product performance, detect anomalies in the service

# due to things like power failure, and more.

# Crucial role in an IoT ecosystem, enabling it to operate by detecting changes

# in an environment and collecting valuable data.

# Conclusion:-

# Sensors are critical to IoT deployments, they are software’s equivalent to our

# eyes and ears. In some cases, an IoT deployment is entirely limited by the

# sensors that are employed. The IoT deployment dictates which sensor is the

# best to choose and then the sensor dictates other hardware and software

# choices.

**CHAPTER 5**

**Python**

* 1. **INTRODUCTION**

When Guido Van Rossum began implementing Python, he was also reading the published scripts from “Monty Python's Flying Circus”, a BBC comedy series from the 1970s. Van Rossum thought he needed a name that was short, unique, and slightly mysterious, so he decided to call the language Python.

Python is a dynamic, interpreted (bytecode-compiled) language. There are no type declarations of variables, parameters, functions, or methods in source code. This makes the code short and flexible, and you lose the compile-time type checking of the source code. Python tracks the types of all values at runtime and flags code that does not make sense as it runs.

An excellent way to see how Python code works is to run the Python interpreter and type code right into it. If you ever have a question like, "What happens if I add an int to a list?" Just typing it into the Python interpreter is a fast and likely the best way to see what happens. (See below to see what really happens!)

$ python        ## Run the Python interpreter  
Type "help", "copyright", "credits" or "license" for more information.  
>>> a = 6       ## set a variable in this interpreter session  
>>> a           ## entering an expression prints its value  
6  
>>> a + 2  
8  
>>> a = 'hi'    ## 'a' can hold a string just as well  
>>> a  
'hi'  
>>> len(a)      ## call the len() function on a string  
2  
>>> a + len(a)  ## try something that doesn't work  
Traceback (most recent call last):  
  File "", line 1, in   
TypeError: can only concatenate str (not "int") to str  
>>> a + str(len(a))  ## probably what you really wanted  
'hi2'  
>>> foo         ## try something else that doesn't work  
Traceback (most recent call last):  
  File "", line 1, in   
NameError: name 'foo' is not defined  
>>> ^D          ## type CTRL-d to exit (CTRL-z in Windows/DOS terminal)

The two lines python prints after you type python and before the >>> prompt tells you about the version of python you're using and where it was built. As long as the first thing printed is "Python 3.", these examples should work for you.

As you can see above, it's easy to experiment with variables and operators. Also, the interpreter throws, or "raises" in Python parlance, a runtime error if the code tries to read a variable that has not been assigned a value. Like C++ and Java, Python is case sensitive so "a" and "A" are different variables. The end of a line marks the end of a statement, so unlike C++ and Java, Python does not require a semicolon at the end of each statement. Comments begin with a '#' and extend to the end of the line.

**Python source code**

Python source files use the ".py" extension and are called "modules." With a Python module hello.py, the easiest way to run it is with the shell command "python hello.py Alice" which calls the Python interpreter to execute the code in hello.py, passing it the command line argument "Alice". See the official docs page on all the different options you have when running Python from the command-line.

Here's a very simple hello.py program (notice that blocks of code are delimited strictly using indentation rather than curly braces — more on this later!):

#!/usr/bin/env python  
  
# import modules used here -- sys is a very standard one  
import sys  
  
# Gather our code in a main() function  
def main():  
    print('Hello there', sys.argv[1])  
    # Command line args are in sys.argv[1], sys.argv[2] ...  
    # sys.argv[0] is the script name itself and can be ignored  
  
# Standard boilerplate to call the main() function to begin  
# the program.  
if \_\_name\_\_ == '\_\_main\_\_':  
    main()

Running this program from the command line looks like:

$ python hello.py Guido

Hello there Guido

$ ./hello.py Alice ## without needing 'python' first (Unix)

Hello there Alice

**Imports, Command-line arguments, and len():-**

The outermost statements in a Python file, or "module", do its one-time setup — those statements run from top to bottom the first time the module is imported somewhere, setting up its variables and functions. A Python module can be run directly — as above "python hello.py Bob" — or it can be imported and used by some other module. When a Python file is run directly, the special variable "\_\_name\_\_" is set to "\_\_main\_\_". Therefore, it's common to have the boilerplate if \_\_name\_\_ ==... shown above to call a main() function when the module is run directly, but not when the module is imported by some other module.

In a standard Python program, the list sys.argv contains the command-line arguments in the standard way with sys.argv[0] being the program itself, sys.argv[1] the first argument, and so on. If you know about argc, or the number of arguments, you can simply request this value from Python with len(sys.argv), just like we did in the interactive interpreter code above when requesting the length of a string. In general, len () can tell you how long a string is, the number of elements in lists and tuples (another array-like data structure), and the number of key-value pairs in a dictionary.

**User-defined Functions**

Functions in Python are defined like this:

# Defines a "repeat" function that takes 2 arguments.  
def repeat(s, exclaim):  
    """  
    Returns the string 's' repeated 3 times.  
    If exclaim is true, add exclamation marks.  
    """  
  
    result = s + s + s # can also use "s \* 3" which is faster (Why?)  
    if exclaim:  
        result = result + '!!!'  
    return result

Notice also how the lines that make up the function or if-statement are grouped by all having the same level of indentation. We also presented 2 different ways to repeat strings, using the + operator which is more user-friendly, but \* also works because it's Python's "repeat" operator, meaning that '-' \* 10 gives '----------', a neat way to create an onscreen "line." In the code comment, we hinted that \* works faster than +, the reason being that \* calculates the size of the resulting object once whereas with +, that calculation is made each time + is called. Both + and \* are called "overloaded" operators because they mean different things for numbers vs. for strings (and other data types).

The def keyword defines the function with its parameters within parentheses and its code indented. The first line of a function can be a documentation string ("docstring") that describes what the function does. The docstring can be a single line, or a multi-line description as in the example above. (Yes, those are "triple quotes," a feature unique to Python!) Variables defined in the function are local to that function, so the "result" in the above function is separate from a "result" variable in another function. The return statement can take an argument, in which case that is the value returned to the caller.

Here is code that calls the above repeat() function, printing what it returns:

def main():  
    print(repeat('Yay', False))      ## YayYayYay  
    print(repeat('Woo Hoo', True))   ## Woo HooWoo HooWoo Hoo!!!

At run time, functions must be defined by the execution of a "def" before they are called. It's typical to def a main() function towards the bottom of the file with the functions it calls above it.

**Indentation**

One unusual Python feature is that the whitespace indentation of a piece of code affects its meaning. A logical block of statements such as the ones that make up a function should all have the same indentation, set in from the indentation of their parent function or "if" or whatever. If one of the lines in a group has a different indentation, it is flagged as a syntax error.

Python's use of whitespace feels a little strange at first, but it's logical and I found I got used to it very quickly. Avoid using TABs as they greatly complicate the indentation scheme (not to mention TABs may mean different things on different platforms). Set your editor to insert spaces instead of TABs for Python code.

A common question beginners ask is, "How many spaces should I indent?" According to the official Python style guide (PEP 8), you should indent with 4 spaces. (Fun fact: Google's internal style guideline dictates indenting by 2 spaces!)

**Code Checked at Runtime:-**

Python does very little checking at compile time, deferring almost all type, name, etc. checks on each line until that line runs. Suppose the above main() calls repeat() like this:

def main():  
    if name == 'Guido':  
        print(repeeeet(name) + '!!!')  
    else:  
        print(repeat(name))

The if-statement contains an obvious error, where the repeat() function is accidentally typed in as repeeeet(). The funny thing in Python ... this code compiles and runs fine so long as the name at runtime is not 'Guido'. Only when a run actually tries to execute the repeeeet() will it notice that there is no such function and raise an error. There is also a second error in this snippet. name wasn't assigned a value before it is compared with 'Guido'. Python will raise a 'NameError' if you try to evaluate an unassigned variable. These are some examples demonstrating that when you first run a Python program, some of the first errors you see will be simple typos or uninitialized variables like these. This is one area where languages with a more verbose type system, like Java, have an advantage ... they can catch such errors at compile time (but of course you have to maintain all that type information ... it's a tradeoff).

**Variable Names**

Since Python variables don't have any type spelled out in the source code, it's extra helpful to give meaningful names to your variables to remind yourself of what's going on. So use "name" if it's a single name, and "names" if it's a list of names, and "tuples" if it's a list of tuples. Many basic Python errors result from forgetting what type of value is in each variable, so use your variable names (all you have really) to help keep things straight.

As far as actual naming goes, some languages prefer underscored parts for variable names made up of "more than one word," but other languages prefer camel Casing. In general, Python prefers the underscore method but guides developers to defer to camel Casing if integrating into existing Python code that already uses that style. Readability counts.

As you can guess, keywords like 'if' and 'while' cannot be used as variable names — you'll get a syntax error if you do. However, be careful not to use built-ins as variable names. For example, while 'str', 'list' and 'print' may seem like good names, you'd be overriding those system variables. Built-ins are not keywords and thus, are susceptible to inadvertent use by new Python developers.

**Features of Python**

There are many features in Python, some of which are discussed below as follows:

**1. Free and Open Source**

[Python](https://www.geeksforgeeks.org/python-programming-language/)language is freely available at the official website and you can download it from the given download link below click on the **Download Python** keyword. [Download Python](https://www.python.org/downloads/) Since it is open-source, this means that source code is also available to the public. So you can download it, use it as well as share it.

**2. Easy to code**

Python is a [high-level programming language](https://www.geeksforgeeks.org/difference-between-high-level-and-low-level-languages/). Python is very easy to learn the language as compared to other languages like C, C#, Javascript, Java, etc. It is very easy to code in the Python language and anybody can learn Python basics in a few hours or days. It is also a developer-friendly language.

**3. Easy to Read**

As you will see, learning Python is quite simple. As was already established, Python’s syntax is really straightforward. The code block is defined by the indentations rather than by semicolons or brackets.

**4. Object-Oriented Language**

One of the key features of [Python is Object-Oriented programming](https://www.geeksforgeeks.org/python-oops-concepts/). Python supports object-oriented language and concepts of classes, object encapsulation, etc.

**5. GUI Programming Support**

Graphical User interfaces can be made using a module such as PyQt5, PyQt4, wxPython, or Tk in python. PyQt5 is the most popular option for creating graphical apps with Python.

**6. High-Level Language**

Python is a high-level language. When we write programs in Python, we do not need to remember the system architecture, nor do we need to manage the memory.

**7. Extensible feature**

Python is an **Extensible** language. We can write some Python code into C or C++ language and also we can compile that code in C/C++ language.

**8. Easy to Debug**

Excellent information for mistake tracing. You will be able to quickly identify and correct the majority of your program’s issues once you understand how to interpret Python’s error traces. Simply by glancing at the code, you can determine what it is designed to perform.

**9. Python is a Portable language**

Python language is also a portable language. For example, if we have Python code for windows and if we want to run this code on other platforms such as Linux, Unix, and Mac then we do not need to change it, we can run this code on any platform.

**10. Python is an Integrated language**

Python is also an Integrated language because we can easily integrate Python with other languages like C, C++, etc.

**11. Interpreted Language:**

Python is an Interpreted Language because Python code is executed line by line at a time. Like other languages C, C++, Java, etc. there is no need to compile Python code this makes it easier to debug our code. The source code of Python is converted into an immediate form called **bytecode**.

**12. Large Standard Library**

Python has a large standard library that provides a rich set of modules and functions so you do not have to write your own code for every single thing. There are many libraries present in Python such as regular expressions, unit-testing, web browsers, etc.

**13. Dynamically Typed Language**

Python is a dynamically-typed language. That means the type (for example- int, double, long, etc.) for a variable is decided at run time not in advance because of this feature we don’t need to specify the type of variable.

**14. Frontend and backend development**

With a new project py script, you can run and write Python codes in HTML with the help of some simple tags <py-script>, <py-env>, etc. This will help you do frontend development work in Python like javascript. Backend is the strong forte of Python it’s extensively used for this work cause of its frameworks like Django and Flask.

**15. Allocating Memory Dynamically**

In Python, the variable data type does not need to be specified. The memory is automatically allocated to a variable at runtime when it is given a value. Developers do not need to write int y = 18 if the integer value 15 is set to y. You may just type y=18.



Source: - <https://bigdata-world.net/python-features/>

Features of Python

# Figure 5.1

# Chapter 6

# Project- Home Automation System

# 6.1 OBJECTIVE

# Our objective is to automate the home to do task more efficiently.

# To Control the home appliances by using computer technology.

# To enhance security.

# To Save electricity

# Our system would be able to detect the person.

# Automatically turn on or off the light.

# It can automatically generate a sound and message the owner in case of fire accident at home.

# Automatic Fan System (Turn On/Off)

# 

# Figure 6.1

# Source: - <https://www.archute.com/10-benefits-automating-home>

# 6.2 Scope:-

# The scope of IoT-based home automation increases by the day because it is a growing technology.

# Integrating smart home devices:-

# Small things at home will be controlled through voice commands and smartphones. This will make daily life simpler and accurate. Spotify songs can be played by voice command. Google has come up with two new home speakers, Home Max and Home Mini.

# Automation outside homes:-

# Sensors will inform people if parking space is available or not, irrespective of where they are. Additional security will also be available with AI implements. Sensors may also be used for streetlights.

# Smart devices:-

# Smart mirrors can be developed that play music, TV may be used for social media, while the fridge may sense outside temperature and work accordingly. Similarly, drones may be used to deliver packages at the right time.

# 6.3 System Requirements:-

# HARDWARE REQUIREMENT

# ● Pentium IV processor or higher

# ● 128 MB RAM (or above)

# ● 40 GB or more HARDDISK

# ● Mouse/Keyboard

# Raspberry pi

# Sensors

# Buzzer

# LDR Circuit

# LCD Display

# ADC Module

# SOFTWARE REQUIREMENT

# ● OS-Windows 7/8/10

# ● Python Interpreter

# ● Proteus 8 Professional

# 6.4 Technology Used:-

# Python (Program)

# Proteus 8 Professional (Simulation Software)

# Raspberry pi

# Libraries Used:-

# Spidev (Library)

# Rpi.GPiO

# Pio (Programmable input output pins)

# Ports

# LCD

# Sensors

# PIR Sensor (Passive infrared sensors to detect heat energy.)

# LDR Sensor (Light Dependent Sensor)

# ADC Module

# Buzzer

# 6.5 What is Raspberry PI?

# The Raspberry Pi is a series of credit card sized single board of computers.

# Created by Eben Upton , CEO of Raspberry Pi Foundation.

# Raspberry Pi Foundation’s aim is to “promote the study of computer science”.

# Raspberry Pi is a low-cost mini-computer with the physical size of a credit card.

# Raspberry Pi runs various ﬂavors of Linux and can perform almost all the tasks that a normal desktop computer can do.

# Raspberry Pi also allows interfacing sensors and actuators through the general purpose I/O pins.

# Since Raspberry Pi runs the Linux operating system, it supports Python "out of the box".

# 

# Raspberry pi

# Figure 6.2

# 6.6 Workflow:-

# 

# Figure 6.3

# 6.7 Source Code and Zip File:-

# <https://github.com/MIETDevelopers/2020a1r034_Hrithik_Sharma_COM-511_Python_Programming_Lab/tree/main/Internship_Project>

# 

# 6.8 Home\_Automation\_Model:-

# 

# Figure 6.4

# 6.9 Benefits:-

# 1. Appliance Safety and Lighting Control:-

# Through automation, you get the ability to control appliances in your home from any location with the touch of a button. You can control the lighting too. This allows you to ensure that the lights and appliances are turned off when you are not home, to save on electricity. You can also turn on the lights at specific times to make it look like you are home, thus increasing the safety of your home.

# 2. Security through Automated Door Locks:-

# There must have been times when you or your kids rushed out of the house in a hurry and forgot to lock the door. With automated door locks, you can lock your doors with just one touch on your smart device from any place. You will also be alerted whenever someone enters the house allowing you constant monitoring, even when you are away from home.

# 3. Increases Awareness through Security Cameras:-

# Security cameras make your home safer. You cannot be home or monitor everything happening in and around the house at all times, but you can automate the security system to provide the kind of security you desire. You can record clips, detect movements and view the activities around your house.

# 4. Allows You To Adjust Temperature:-

# One of the most unpleasant things is coming to a house that is too warm or too cold. You can bring the temperature to your desired level by controlling the HVAC system but it would take time for the room to reach the desired temperature. With a home automation system in place, you can adjust thermostats from any place so that it is at a comfortable temperature by the time you reach home.

# 5. Saves Time:-

# Life has become really busy these days and various household chores keep you on your toes. Imagine running back home for a few minutes just to adjust some household item or opening the door for your kids after their school ends. You can manage all that through automation without having to go home. This helps in saving a lot of precious time and promotes productivity.

# 6. Convenience and Cost Efficiency:-

# Automation lets you control electrical appliances making sure that they are not wasting power when not in use. It gives you the convenience to control different devices even when you are not home. This helps in reducing electric costs and helps you save money.

# 7. Contributes To Economy:-

# By installing home automation system, you use energy efficiently. This helps in contributing to the economy by utilizing only those ressources that you need.

# 8. Allows You To Be Worry-Free:-

# You don’t have to obsessively worry about minute things at home as you can keep a constant check on the house, irrespective of where you are. Home automation brings with it peace of mind.

# 9. Gives You Control Even When You Are Out Of Town:-

# Forget about leaving the key with your neighbour when you leave town. With automation, you can schedule a time for them to enter your house to perform chores such as watering plants or feeding your pet, or you can let them in and out through your smartphone. This will relieve you from the worry of someone having an unrestrained access to your belongings or them losing your key. You can also make sure that the chores are being completed as requested.

# 10. Let’s You Keep a Check on Your Children:-

# The safety of children is the most important thing for any parent. Home automation helps you keep them safe by keeping tabs on them even when you aren’t home. You can open the doors, and keep the porch lit when they get back home in your absence. You can monitor when they come and go, through security cameras.

# 6.10 Conclusion:-

# Home Automation is undeniably a resource which can make a home environment automated. People can control their electrical devices via these Home Automation devices and set up controlling actions through Mobile. In future this product may have high potential for marketing.

# REFERENCES

1. <https://www.geeksforgeeks.org/introduction-to-internet-of-things-iot-set-1/>
2. <https://www.javatpoint.com/iot-architecture-models>
3. <https://www.javatpoint.com/iot-security-camera-and-door-unlock-system>
4. <https://www.topicsforseminar.com/2018/04/iot-internet-of-things-seminar-report.html>
5. <https://www.simplilearn.com/iot-devices-article>
6. <https://www.techtarget.com/iotagenda/definition/IoT-device>
7. <https://internetofthingsagenda.techtarget.com/definition/RFID-radio-frequency-identification>
8. <https://internetofthingsagenda.techtarget.com/definition/IoT-attack-surface>
9. <https://internetofthingsagenda.techtarget.com/tip/Internet-of-Things-IOT-Seven-enterprise-risks-to-consider>
10. <https://www.zipitwireless.com/blog/what-are-iot-sensors-types-uses-and-examples>
11. <http://docs.python.org/using/cmdline>
12. <http://python.org/dev/peps/pep-0008/#indentation>
13. <https://www.geeksforgeeks.org/python-features/>
14. <https://www.archute.com/10-benefits-automating-home/>