1. INTRODUCTION

1.1 Purpose of the system:

Visual deficiency is a condition of without the visual recognition because of physiological or neurological elements .Vision is the most excellent blessing and it a path at how a man takes a gander at his environment yet Many individuals experience the ill effects of genuine visual debilitations keeping them from going without help . Consequently, they have to utilize an extensive variety of apparatuses and procedures to help them in their portability. There is a worldwide image device of visually impaired and outwardly disabled individuals simply like the white stick with a red tip which is utilized to improve the visually impaired development .However, this apparatus has a few requirements: in length of the stick, restrictions in perceiving impediments, and furthermore trouble to keep it in broad daylight places. As of late, numerous strategies have been produced to improve the portability of visually impaired individuals that depend on flag preparing and sensor innovation. These called Electronic Travel Aid (ETA) gadgets help the ignorant concerning move openly in a situation paying little mind to its dynamic changes. ETAs are principally ordered into two noteworthy angles: sonar input (laser flag, infrared signs, or ultrasonic signs) and camera input frameworks (comprises essentially of a smaller than usual CCD camera) . The way these gadgets work simply like the radar framework that utilizations ultrasonic fascicle or laser to recognize tallness, the heading, and speed of settled and moving items. The separation between the individual and the impediments is measured when of the wave travel. In any case, every single existing framework advise the incognizant in regards to the nearness of a question at a particular separation before or close to him/her through tone signals or potentially vibrations that need to preparing. These points of interest allow the ignorant concerning change his/her way just, however they are not happy and safe. Our work offers a straightforward, proficient, configurable electronic direction framework for both visually impaired and outwardly impeded individuals to help them in their versatility paying little respect to where they are, open air or indoor. Moreover, the client of the framework does not have to convey a stick or other stamped device. He/she can simply wear a cap and hand scaled down stick (size of a pen) simply like others. The proposed framework uses four ultrasonic sensors (three in cap and one in pen) convey the reflected signs to a microcontroller to deliver diverse voice messages through headphone in light of the sensors yield to educate the visually impaired about the question points of interest. Furthermore, the proposed framework utilizes GPS and

GSM modules to empower the oblivious in regards to speak with the home or any favored portable number amid crisis and follow up him/her. Likewise, it empowers the incognizant in regards to be frightened or a favored cell phone on the off chance that he/she is outside a specific region.

1.2 Existing System:

There is no specific system at present which helps the blind or impaired much. The only current system available at the moment is the Braille script. The script is an alphabet (phonetic) used by the blind people for writing and reading. Each Braille cell or character is made up of 6 dots, arranged in the form of matrix of dots. A dot may be raised or flat, giving a total of possible combinations for 6 dots. Blind people use touch to read the raised dots.
The disadvantage of this system is that it really can't tell or narrate the visually impaired person the description of the scene in front of him.It's advantage is only for the literates to read and write but not the description of Scene.

Disadvantages:

1. There is no specific system atpresent which helps the blind or impaired much.

2. The only current system available atthe moment is the Braille script.

1.3 Proposed System:

The system we are proposing is a very advanced system known as the Smart Cap which makes use of technology known as Internet of Things using raspberry pi platform. The proposed system makes use of a webcam which then makes use of raspberry-pi and a text-to-speech converter(TTS) to convert the images captured into the form of sound i.e. audio output. The proposed system has many advantages. It includes the feature of safety which is most important to an human life, especially to visually impaired persons since they can't see what's coming or happening in front of them. The proposed system consists of webcam retrofitted on cap of visually impaired which narrates the description of scene. Through this system, the user will be knowing what's happening in front of them and can take appropriate actions. Suppose there is a car approaching towards the person, then an audio output saying that there is 1 car in front of you will be heard by person and hence can avoid the collision by being careful

Advantages:

1. It gives a new perspective of life for the visually impaired.
2. Navigation made safer and easier for the blind.

2. SYSTEM REQUIREMENTS

2.1 H/W System Configuration:-

- Raspberry Pi
- Webcam
- Monitor
- SDcard
- Charger
- Mouse
- Keyboard
- Ethernet
- 3.5mm Audiojack Earphones

2.2 S/W System Configuration:-

- Opencv
- VNC server
- Python

3. SYSTEM STUDY

FEASIBILITY STUDY

The possibility of the venture is dissected in this stage and business proposition is advanced with an extremely broad arrangement for the venture and some cost gauges. Amid framework investigation the possibility investigation of the proposed framework is to be completed. This is to guarantee that the proposed framework is not a weight to the organization. For practicality investigation, some comprehension of the real necessities for the framework is basic.

Three key contemplations required in the plausibility investigation are

3.1 ECONOMICAL FEASIBILITY
3.2 TECHNICAL FEASIBILITY
3.3 SOCIAL FEASIBILITY

3.1 ECONOMICAL FEASIBILITY

This review is completed to check the monetary effect that the framework will have on the association. The measure of store that the organization can fill the innovative work of the framework is constrained. The uses must be defended. Accordingly the created framework too inside the financial plan and this was accomplished on the grounds that a large portion of the innovations utilized are unreservedly accessible. Just the altered items must be obtained.

3.2 TECHNICAL FEASIBILITY

This audit is done to check the particular plausibility, that is, the specific necessities of the structure. Any structure made must not have an interest on the available particular resources. This will provoke levels of notoriety on the available specific resources. This will provoke levels of ubiquity being determined to the client. The made system must have an unassuming essential, as simply irrelevant or invalid changes are required for executing this structure.

3.3 SOCIAL FEASIBILITY

The part of study is to check the level of acknowledgment of the framework by the client. This incorporates the way toward preparing the client to utilize the framework effectively. The client must not feel debilitated by the framework, rather should acknowledge it as a need.

The level of acknowledgment by the clients exclusively relies on upon the techniques that are utilized to instruct the client about the framework and to make him acquainted with it. His level of certainty must be raised with the goal that he is likewise ready to make some valuable feedback, which is invited, as he is the last client of the framework.

4.SOFTWARE ENVIRONMENT

4.1 Internet of Things

Introduction

Internet0fThings(I0T) comprises 0f things that have unique identities, which are connected to the web. While many existing devices, such as networked computers or 4G-enabled mobile phones, already have some form of unique identities and are also connected to the Internet, the focus on IoT is in the configuration, control and networking via the Internet of devices or "things" that are traditionally not associated with the internet. These include devices such as thermostats, utility meters, a Bluetooth connected headset, irrigation pumps and sensors, or control-circuits for an electric car's engine. Internet of Things is anew revolution in the capabilities of the end points that are connected to the Internet, and is driven by the advancement in capabilities(in combination with lower costs) in mobile devices, wireless communications, sensor networks, cloud technologies and networking. Experts forecast that by the year 2020 there will be a total of 50 billion devices/ things connected to the Internet. Therefore, the major industry players are excited by the prospects of new markets for their products. The products include hardware and software components for IoT endpoints, hubs, or control centers of the IoT universe.

The scope of IoT is not limited to just connecting things (devices, appliances, machines) to the Internet. IoT allows these things to communicate and exchange data (Control& information, that could include data associated with users) while executing meaningful application towards a common user or machine goal. Data itself does not have a meaning until it is contextualized processed into useful information. Applications on IoT networks extract and create information from lower level data by filtering, processing, categorizing, condensing and contextualizing the data. This information obtained is then organized and structured to infer knowledge about the system and/or its users, its environment, and its operations and progress towards its objectives, allowing a smarter performance, as shown in Figure 1.1. For example, consider a series of raw sensor measurements ((72,45):(84,56)) generated by a weather monitoring station, which by themselves do not have any meaning or context. To give meaning to the data .a context is added . Which in this example can be that each tuple in data represents the temperature and humidity measured every minute .With this context added we know the meaning(or information)of the measured data tuples .Further information is obtained by categorizing .considering or processing this data .For example the average temperature and humidity readings for last five minutes is obtained by averaging the last five data tuples .The next step is to organize the information and understand the relationships between pieces of information to infer knowledge which can be put into action .For example, an alert is raised if the average temperature in last five minutes exceeds 120F,and this alert may be conditioned on the user's geographical position as well.

The applications of Internet of Things span a wide range of domains including (but not limited to) homes, cities, environment, energy, systems, retail, logistics, industry, agriculture and health as listed .For homes ,IOT has several applications such as smart lighting that adapt the lighting to suit the ambient conditions .smart appliances that can be remotely monitored and controlled, intrusion detection systems ,smart smoke detectors ,etc. For cities, IOT has applications such as smart parking systems that provide status updates on available slots ,smart lighting that helps in saving energy ,smart roads that provide information on driving conditions and structural health monitoring systems .For environment ,IOT has applications such as weather monitoring , air and noise pollution , forest fire detection and river flood detection systems . For energy systems, IOT has applications such as including smart grids , grid integration of renewable energy sources and prognostic health management systems . For retail domain, IOT has applications such as inventory management , smart payments and smart vending machines. For agriculture domain, IOT has applications such as smart irrigation systems that help in saving water while enhancing productivity and green house control systems .Industrial applications such as smart irrigation systems. Industrial applications of IOT include machine diagnostics and prognosis systems. For health and lifestyle, IOT has applications such as health and fitness monitoring systems and wearable electronics.

Definition & Characteristics of IOT

Definition: A dynamic worldwide system foundation with self-arranging abilities in view of standard and interoperable correspondence conventions where physical and virtual thing have characters, physical characteristics, and virtual identities and utilize keen interfaces, and are consistently coordinated into the data organize ,often communicate data associated with users and their environments.

Let us examine this definition of IOT further to put some of the terms into perspective.

Dynamic & Self-Adapting: IOT devices and systems may have the capability to dynamically adapt with the changing contexts and take actions based on their operating conditions . user's context ,or sensed environment. For example, consider surveillance system compromising of a number of surveillance cameras. The surveillance cameras day or night. Cameras could switch from lower resolution modes when any motion is detected and alert nearby cameras to do the same. In this example, the surveillance system is adapting itself based on the context and changing(e.g., dynamic)conditions.

Self-Configuring: IOT devices may have self-configuring capability, allowing a large number of devices to work together to provide certain functionality(such as weather monitoring). These devices have the ability configure themselves(in association with the IOT infrastructure), setup the networking, and fetch latest software upgrades with minimal manual or user intervention.

Interoperable Communication Protocols: IOT devices may support a number of interoperable communication protocols and communicate with other devices and also with the infrastructure. We describe some of the commonly used communication protocols and models in later sections.

Unique Identity: Each IOT devices has a unique identity and a unique identifier (Such as an IP address or a URI). IOT systems ,may have intelligent interfaces which adapt based on the context, allow communicating with users and the environmental contexts. IOT device interfaces allow users to query the devices, monitor their status, and control them remotely, in association with the control, configuration and management infrastructure.

Integrated into Information Network: IOT devices are usually integrated into the information network that allows them to communicate and exchange data with other devices and systems. IOT devices can be dynamically discovered in the network, by other devices and/or the network, and have the capability discovered in the network, and have the capability to describe themselves(and their devices or user applications. For example, a weather monitoring node can describe its monitoring capabilities to another connected node so that they can describe its monitoring capabilities to another connected node so that they can communicate and exchange data. Integration in to the information network helps in making IOT systems" smarter" due to the collective intelligence of the individual devices in collaboration with the infrastructure. thus , the data from a large number of connected weather monitoring IOT nodes can be aggregated and *analyzed to predict the wealth.

Physical Design of IOT

Things in IOT

The "Things" in IOT usually refers to IOT gadgets which have interesting characters and can perform remote detecting, impelling and observing abilities. IOT gadgets can trade information with other associated gadgets and applications (straightforwardly or in a roundabout way), or gather information from different gadgets and process the information either locally or send the information to brought together servers or cloud-based application back-finishes for preparing the information, or play out a few errands locally and different assignments inside the IOT framework, in view of worldly and space requirements (i.e., memory, handling capacities, correspondence latencies and rates, and due dates).

Figure1.2.1 shows a block diagram of a typical IOT device. An IOT device may consist of several interfaces for connections to other devices, both wired and wireless. these include (i) I/O interfaces for sensors, (ii) interfaces for internet connectivity, (iii) memory and storage interfaces and (iv) audio/video interfaces. An IOT device can collect various types of data from the on-board or attached sensors, such as temperature, humidity, light intensity. The sensed data can be connected actuators that allow them to interact with other physical entities (including non-IOT devices and systems) in the vicinity of the device. For example, a relay switch connected to an IOT device can turn an appliance on/off based on the commands sent to the IOT device over the Internet.

IOT Protocols

Link Layer

Link Layer protocols determine how the data is physically sent over the network's physical layer or medium(e. g., copper wire, coaxial a cable, or a radio wave). The scope of the link layer is the network connection to which host is attached. Hosts on the same link exchange data packets over the link layer using link layer protocols. Link layer determines how the packets are coded and signaled by the hardware device over the medium to which the host is attached (such as a coaxial cable). Let us now look at some link layer protocols which are relevant in the context of IOT.

- **802.3-ETHERNET:** IEEE 802.3 is a collection of wired Ethernet standards for the link layer.

  For example, 802.3 is the standard for 10BASIES Ethernet that uses coaxial cable as a shared medium, 802.3.i is the standard for 10BASE-T Ethernet over copper twisted-pair connections, 802.3ae is the standard for 10Gbit/Ethernet over fiber, and so on. optic connections, These standards provide data rates from 10Mbs to40Gbs and higher. The shared medium in Ethernet can be a coaxial cable, twisted-pair wire or an optical fiber. the shared medium(i. e., broadcast medium)carries the communication for all the devices propagation conditions and transceiver capabilities. The specifications of the 802.3 standards are available on the IEEE802.3 working group website .
- **802.3-WIFI:** IEEE802.11 is a collection of wireless local area network(WLAN)communication standards, including extensive description of the link layer. For example, 802.11a operates in the 5GHZ band, 802.11b and 802.11g operate in the 2.4GHZ band 802.11n operates in the2.4/5GHZ bands, 802.11ac operates from 1MB/S to upto6.75Gb/s. The specifications of the 802.11 standards are available on the IEEE802.11 working group website
-

802.11-WIFI:IEEE802.16 is a collection of wireless broadband standards, including extensive description for the link layer(also called WiMax). WiMax.standards provide data rates from 1.5 Mb/s to 1 G b/s . The recent update(802.16m) provides data rates of 100Mbit/s for mobile stations and 1 G bit/s for fixed stations .The specifications of the 802.11.standards are readily available on the IEEE802.16 working group website

- **802.15.4--LR-WPAN**: IEEE 802.15.4 is a collection of standards for low-rate wireless personal area networks(LR-WPANs). These standards form the basis of specifications for high level communication protocols such as Zig Bee. LR-WPAN standards provide data rates from 4Kb/s 250 Kb/s. These standards provide low-cost and low-speed communication for power constrained devices. The specifications of the 802.15.4 standards are available on the IEEE802.15 working group website.

- **2G/3/4G-MobileCommunication**: There are different generations of mobile communication standards including second generation (2G including GSM and CDMA), third generation(3G-including UMTS and CDMA2000) and fourth generation(4G-including LTE). IOT devices based on these standards can communicate over cellular networks. Data rates for these standards range for 9.6 K b/s(for 2G) to up to 100Mb/s(for 4G) and are available from the 3GPP websites.

Network/Internet Layer

The network layers are responsible for sending of IP data grams from the source network to the destinations network. This layer performs the host addressing and packet routing. The data grams contain the source and destination addresses which are used to route them from the source to destination across multiple networks. Host identification is done using hierarchical IP addressing schemes such as IPV4 or IPV6.

- **IPV4:** Internet Protocol version 4(IPV4) is the most deployed Internet Protocol the is used to identify the devices on a network using a hierarchical addressing scheme. IPV4 uses a 32-bit address scheme that allows total of 4,294,967,296 addresses. As more and more devices got connected to the Internet, these addresses got exhausted in the year 201. IPV4 has been succeeded by IPV6. The IP protocols establish connections on packet network .but do not guarantee delivery of packets. Guaranteed delivery and formally described in RFC791.
- **IPV6:** Internet Protocols version 6(IPV6) IS The newest version of Internet Protocols and successor to IPV4. IPV6 uses 128-bit address scheme that allows total of 3.4*10 38 address.IPV4 is formally described in RFC 2460 .

Transport Layer

The transport layer protocols provide end-end message transfer capability independent of the underlying network. The message transfer capability can be set up on connections, either using handshakes(as in TCP) or without handshakes/acknowledgements(as in UDP).The transport layer provides function such as error control, segmentation, flow control and congestion control.

- **TCP:** TCP (also called Transmission Control Protocol) is the most widely used protocol . that is used by web browsers along with HTTP,HTTPS application layer protocols, email programs(SMTP application layer protocols)and file transfer(FTP). TCP is connection oriented and stateful protocol. While IP protocol deals with sending packets. TCP ensures reliable transmission of packets in-order. TCP also provides error detection capability so that duplicate packets can be discarded and lost packets are retransmitted. The flow control capability of TCP ensures that rate at which the sender sends the data is not to high for the receiver to process. The congestion collapse which can lead to degradation of network performance TCP is described in RFC 793.
- **UDP:** Unlike TCP, which requires carrying out an initial setup procedure. UDP of a connectionless protocol. UDP is useful for the time-sensitive applications that have very small data units to exchange and do not stateless protocol.UDP does not provide guaranteed delivery or ensuring connections created are reliable. UDP is described inRFC768.

Application Layer

Application Layer protocols define how the applications interface with the lower layer protocols to send the data over the network. The application data, typically in files, is encoded by the application layer protocol and encapsulated in the transport layer protocol which provides connection or transaction oriented communication over the network. Port numbers are used for application address (for example port 80 for HTTP port 22 for SSH etc.) .Application layer protocols enable process-to-process connections using ports.

- **HTTP:** HTTP (also called Hypertext Transfer Protocol) is the application layer protocol that forms the foundation of the World Wide Web(WWW). HTTP include commands such as GET, PUT, POST, DELETE, HEAD,TRACE, OPTIONS, etc. The protocol follows a request-response model where a client can sends a requests to a server using the HTTP commands. HTTP is a stateless protocol and each HTTP request is independent of the other requests. An HTTP client can be a browser or an application running on the client(e.g., an application running on an IOT device, a mobile application or other software). HTTP protocol uses Universal Resource Identifiers(URLs) to identify HTTP resources. HTTP is described in RFC 2616[11].

- **CoAP:** Constrained Application Protocol (CoAP) is an application layer protocol for machine-to-machine(M2M) applications. meant for constrained environments with constrained devices and constrained networks. Like HTTP,COAP is a web transfer protocol and uses a request-response model, however it runs on top of UDP instead of TCP. COAP uses a client-server architecture where clients communicate with servers using connectionless data grams COAP is designed to easily interface with HTTP. Like HTT[, COAP supports methods such as GET,PUT,POST, and DELETE,COAP draft specifications are available on IEFT Constrained environments(CORE) working Group Website[12].
- **Web Socket:** Web Socket protocol allows full-duplex communication over a single socket connection for sending message between client and server. Web server is based on TCP and allows streams of messages to be sent back and forth between the client and server while keeping the TCP connection open. The client can be a browser, a mobile application or an IOT device. Web Socket is described in RFC6455[13].
- **MQTT:** Message Queue Telemetry Transport (MQTT) is a light-weight messaging protocol based on the publish-subscribe model. MQTT uses a client-server(also called MQTT Broker) and publishes messages to topics. MQTT is well suited for constrained environments where the devices have limited processing and memory resources and the network bandwidth is low MQTT specifications are available on IBM developer Works[14].
- **XMPP:** Extensible Messaging and Presence Protocol(XMPP) is a protocol for real-time communication and streaming XML data between entities. XMPP powers wide range of applications including messaging, presence, data syndication, gaming, multi-party chat and voice/video calls. XMPP allows sending small chunks of XML data from one network entry to another in near real-time. XMPP is a decentralized protocol and uses a client-server architecture. XMPP supports both client-to-server and server-to-server communication paths. In the context of IOT, XMPP allows real-time communication between IOT devices. XMPP is described in RFC6120[15].
- **DDS:** Data Distribution Service(DDS) is a data-centric middleware standard for device-to-device or machine-to-machine. DDS uses a publish-subscribe for model where publishers (e g. devices that generate data) create topics to which subscribers(e. g.,

devices that want to consume data)can subscribe. Publisher is an object responsible for data distribution and the subscriber is responsible for receiving reliability. DDS provides quality-of-service(QOS) control and configurable reliability .DDS is described in Object Management Group(OMG)DDS specification

AMQP: Advanced Messaging Queuing Protocol(AMQP) is an open application layer protocol for business messaging. AMQP supports both point-to-point publishers(e.g., devices or application that generate data). Publishers publish the connections to consumers(application that process data). Publishers publish the messages to exchange which then distribute message copies to queues. Messages are either delivered by the broker to the consumers which have subscribed to the queues or the consumers can pull the messages from the queues. AMQP specification is available on the AMQP working group website[17].

Logical Designs of IoT

IoT Functional Blocks

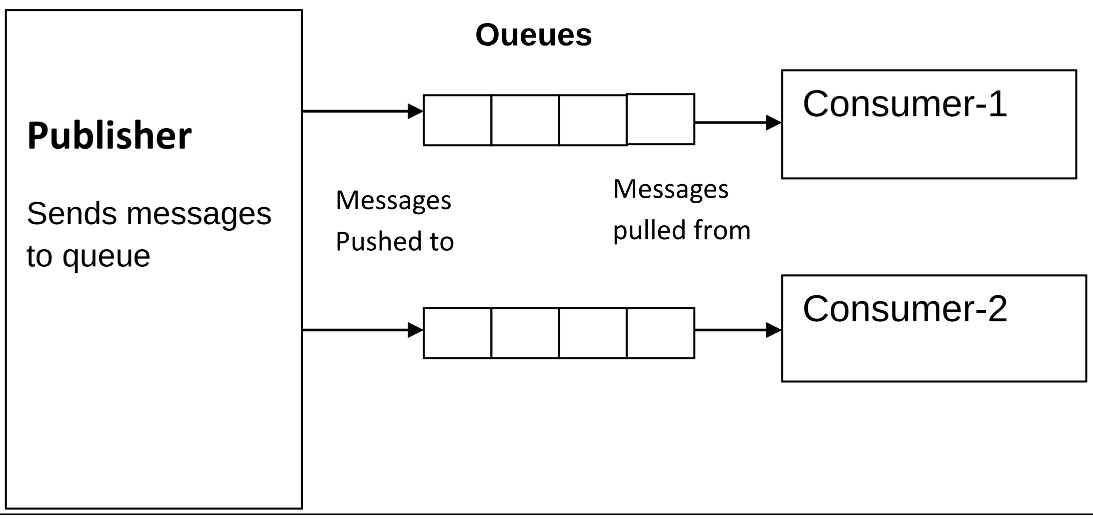
An IoT system comprises of a number of functional blocks that provide the system the capabilities for identification, sensing, actuation, and management as shown in Figure 1.3.1. These functional blocks are described as follows:

- **Device**: An IoT system comprises of devices that provide sensing, actuation, monitoring and control functions. You learned about IoT devices in section 1.2.
- **Communication**: The communication block handles the communication for the Iot system. You learned about various protocols used for communication by IoT systems in section 1.2.
- **Services**: An IoT system uses various types of IoT services such as services for device monitoring, device control services, data publishing services and services for device discovery.
- **Management:** Management functional block provides various functions to govern the IoT system.
- **Security**: Security functional block secures the IoT system and by providing functions such as authentication, authorization, message and content integrity, and data security.

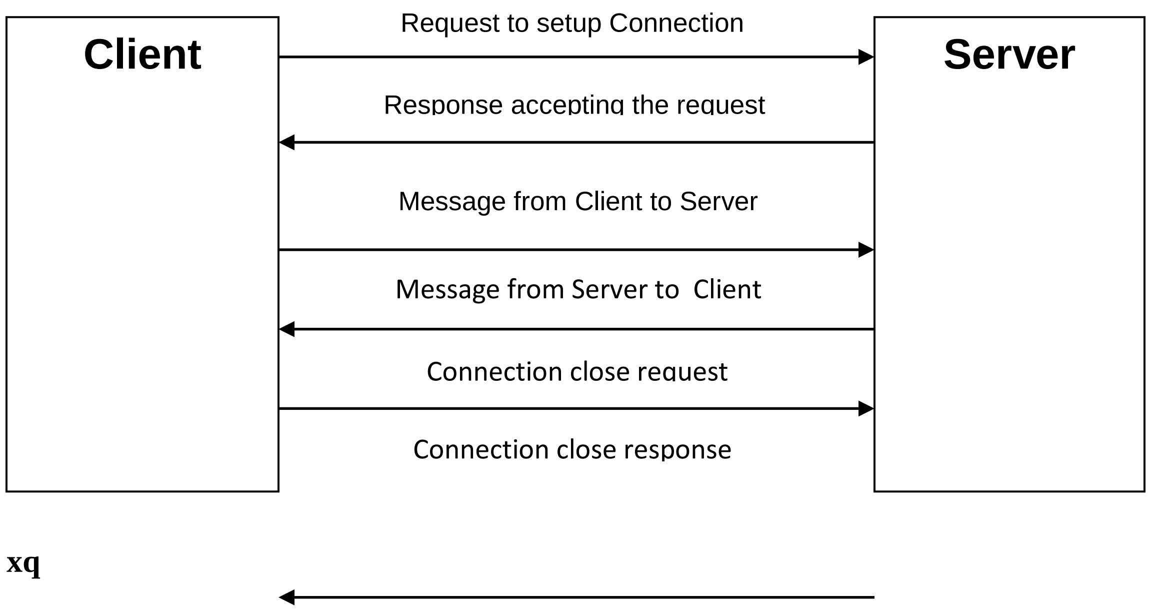
IoT Communication Models

- **Request-Response:** Request-Response is a communication modelin which the client sends request to the server and the server responds to the requests. When the server receives a request, it decided how to respond, fetches the data, retrieves resource representations, prepares the response, and then sends the response to the client. Request- Response model is a stateless communication model and each request-response pair is independent of others. Figure1.3.2(1) shows the client-server interactions in the request-response model.

- **Publish-Subscribe :**Publish-Subscribe is a communication model that involves publishers, brokers and consumers. Publishers are the source of data. Publishers send the data to the topics which are managed by the broker. Publishers are not aware of the consumers. Consumers subscribe to the topics which are managed by the broker. When the broker receives data for a topic from the publisher, it sends the data to all the subscribed consumers. Figure 1.8 shows the publisher-broker consumer interactions in the publish-subscribe model.
- **Push-Pull :** Push-Pull is a communication model in which the data producers push the data to queues and the consumers pull the data from the queues. Producers do not need to be aware of the consumers. Queues help in decoupling the messaging between the producers and consumers. Queues also act as a buffer which helps in situations when there is a mismatch between the rate at which the producers push data and the rate at which the consumers pull data. Figure 1.3.2(3) shows the publisher- queue-consumer interactions in the push-pull model.



- **Exclusive Pair:** Exclusive Pair is a bi-directional, fully duplex communication model that uses a persistent connection between the client and server. Once the connection is setup it remains open until the client sends a request to close the connection. Client and server can send messages to each other after connection setup. Exclusive pair is a stateful communication model and the server is aware of all the open connections. Figure 1.3.2(4)shows the client -server interactions in the exclusive pair model.


Client
Server
Request to setup Connection
Response accepting the request
Message from Client to Server
Message from Server to Client
Connection close request
Connection close response

xq

Figure 4.1.2:Exclusive Pair Communication Model

IoT Communication APIs

REST-based Communication APIs

Representational State Transfer (REST) (88) is a set of architectural principles by which you can design web services and web apis that focus on a system's resources and how resource states are addressed and transferred. REST APIs follow the request-response communication model described in previous section. The REST architectural constraints apply to the components, connectors, and date elements, within a distributed hypermedia system. The REST architectural constraints are as follows:

- **Client-Server:** The principle behind the client-server constraint is the separation of concerns. For example, clients should not be concerned with the storage of data which is a concern of the server. Similarly, the server should not be concerned about the user interface, which is a concern of the client. Separation allows client and server to be independently developed and updated.

- **Stateless:** Each request from client to server must contain all the information necessary to understand the request, and cannot take advantage of any stored context on the server. The session state is kept entirely on the client.

- **Cache-able**: Cache constraint requires that the data within a response to a request be implicitly or explicitly labeled as cache-able or non-cache-able. If a response is cache-able, then a client cache is given the right to reuse that response data for later, equivalent request. Caching can partially or completely eliminate some interactions and improve efficiency and scalability.
- **Layered System:** Layered system constraint, constrains the behavior of components such that each component cannot see beyond the immediate layer with which they are interacting. For example, a client cannot tell whether it is connected directly to the end server, or to an intermediary along the way. System scalability can be improved by allowing intermediaries to respond to request instead of the end server, without the client having to do anything different.

- **Uniform Interface:** Uniform Interface constraint requires that the method of communication between a client and a server must be uniform. Resources are identified in the request (by URIs in web based systems) and are themselves separate from the representations of the resources that are turned to the client. When a client holds a representation of a resource it has all the information required to update or delete the resource (provided the client has required permissions). Each message includes enough information to describe how to process the message.

- **Code on demand**: Servers can provide executable code or scripts for clients to execute in their context. This constraint is the only one that is optional.

A RESful web service is a "web API" implemented using HTTP and REST principles. Figure 1.3.3(1)_ shows the communication between client and server using REST APIs. Figure 1.3.3(2) shows the interactions in the request-response model used by REST.RESTful web service is a collection of resources which are represented by URIs.RestFUL web API has a base URI (e.g.http://example.com/api/tasks/). The client send requests to these URIs using the methods defined by the HTTP protocol (e.g., GET, PUT, POST, or DELETE), as shown in Table 1.1.

A RESTful web service can support various Internet media types (JSON being the most popular media type for RESTful web services). IP for Smart Objects Alliance (IPSO Alliance) has published an Application Framework that defines a RESTful design for use in IP smart object systems.

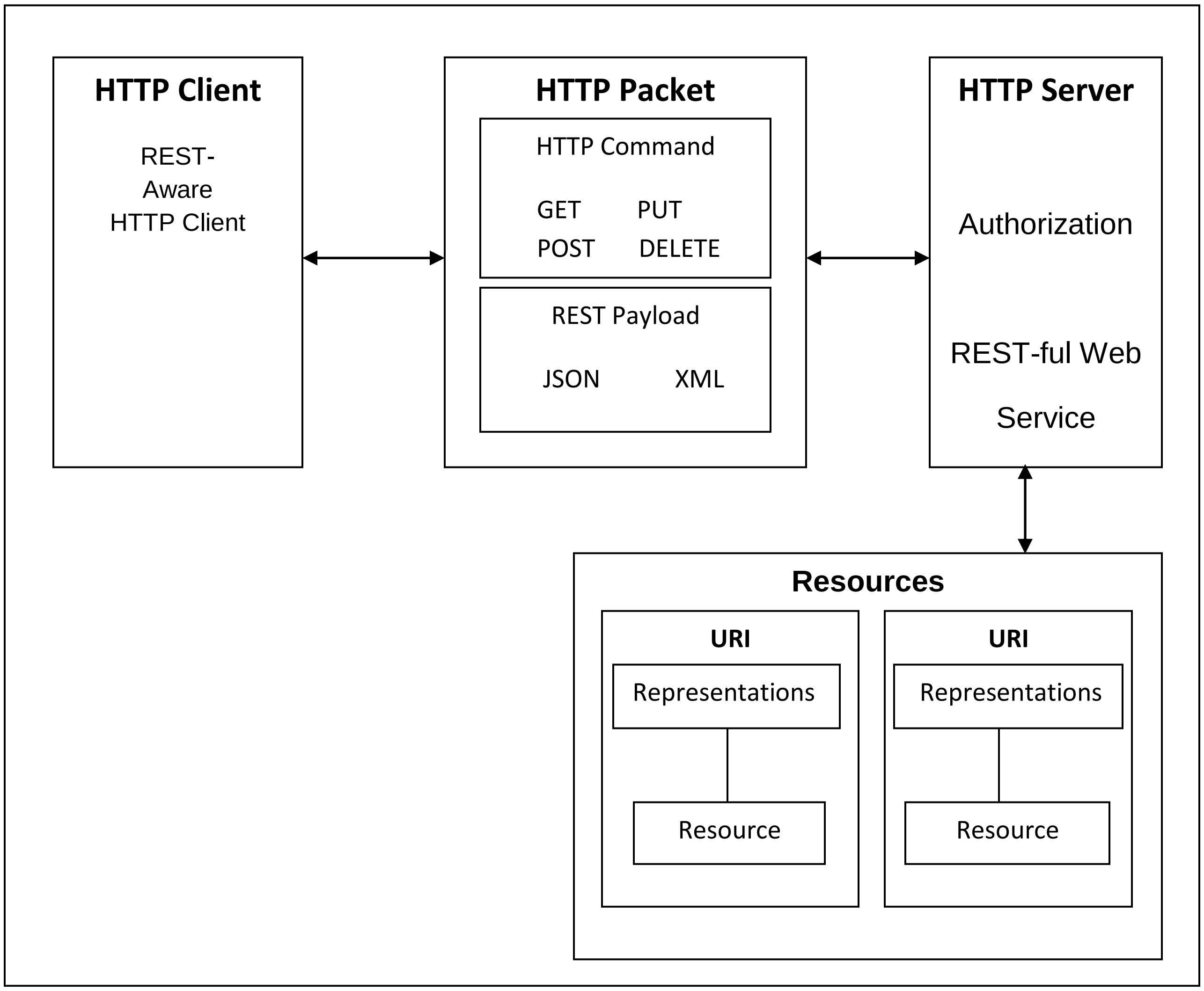


Figure 4.1.3: Communication with REST APIs

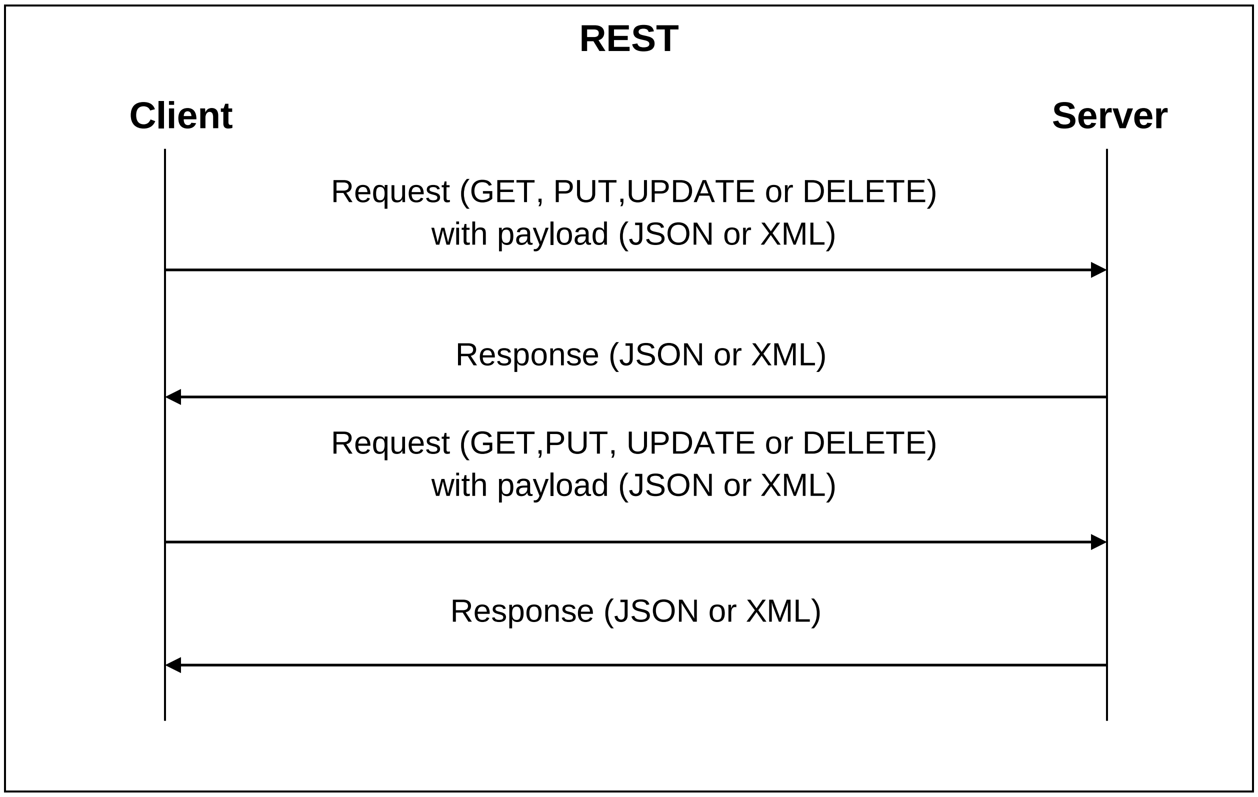

REST
Client
Server
Request (GET, PUT,UPDATE or DELETE) with payload (JSON or XML)
Response (JSON or XML)
Request (GET,PUT, UPDATE or DELETE) with payload (JSON or XML)
Response (JSON or XML)

Figure 4.1.4:Request-response model used by REST

WebSocket-based Communication APIs

Websocket APIs allow bi-directional, full duplex communication between clients and servers. Websocket APIs follow the exclusive pair communication model described in previous and as shown in Figure 1.3.3(3). Unlike request-response APIs such as REST, the WebSocket APIs alow full duplexcommunication and do not require a new connection to be setup for each message to be sent. WebSocket communication begins with a connection setup request sent by the client to the server. This request (called a WebSocket handshake) is sent over HTTP and the server interprets it as an upgrade request. If the server supports WebSocket protocol, the server responds to the WebSocket handshake response. After the connection is setup, the client and server can send data/messages to each other in full-duplex mode. WebSocket APIs reduce the network traffic and latency as there is no overhead for connection setup and termination requests for each message. WebSocket is suitable for IoTapplications that have low latency or high throughput requirements.

HTTP Method	**Resource Type**	**Action**	**Example**
GET	Collection URI	List all the resources in a collection	http://example.com/api/tasks/ (list all tasks)
GET	Element URI	Get information about a resource	http://example.com/api/tasks/1/(get information on task-1)
POST	Collection URI	Create a new resource	http://example.com/api/tasks/(create a new task from data provided in the request)
POST	Element URI	Generally not used	
PUT	Collection URI	Replace the entire collection with another collection	http://example.com/api/tasks/(replace entire collection with data provided in the request)

PUT	Element URI	Update a resource	http://example.com/api/tasks/1/(update task-1 with data provided in the request)
DELETE	Collection URI	Delete the entire collection	http://example.com/api/tasks/(delete all tasks)
DELETE	Element URI	Delete a Resource	http://example.com/api/tasks/1/(delete task-1)

Table 4.1.1: HTTP request methods and actions

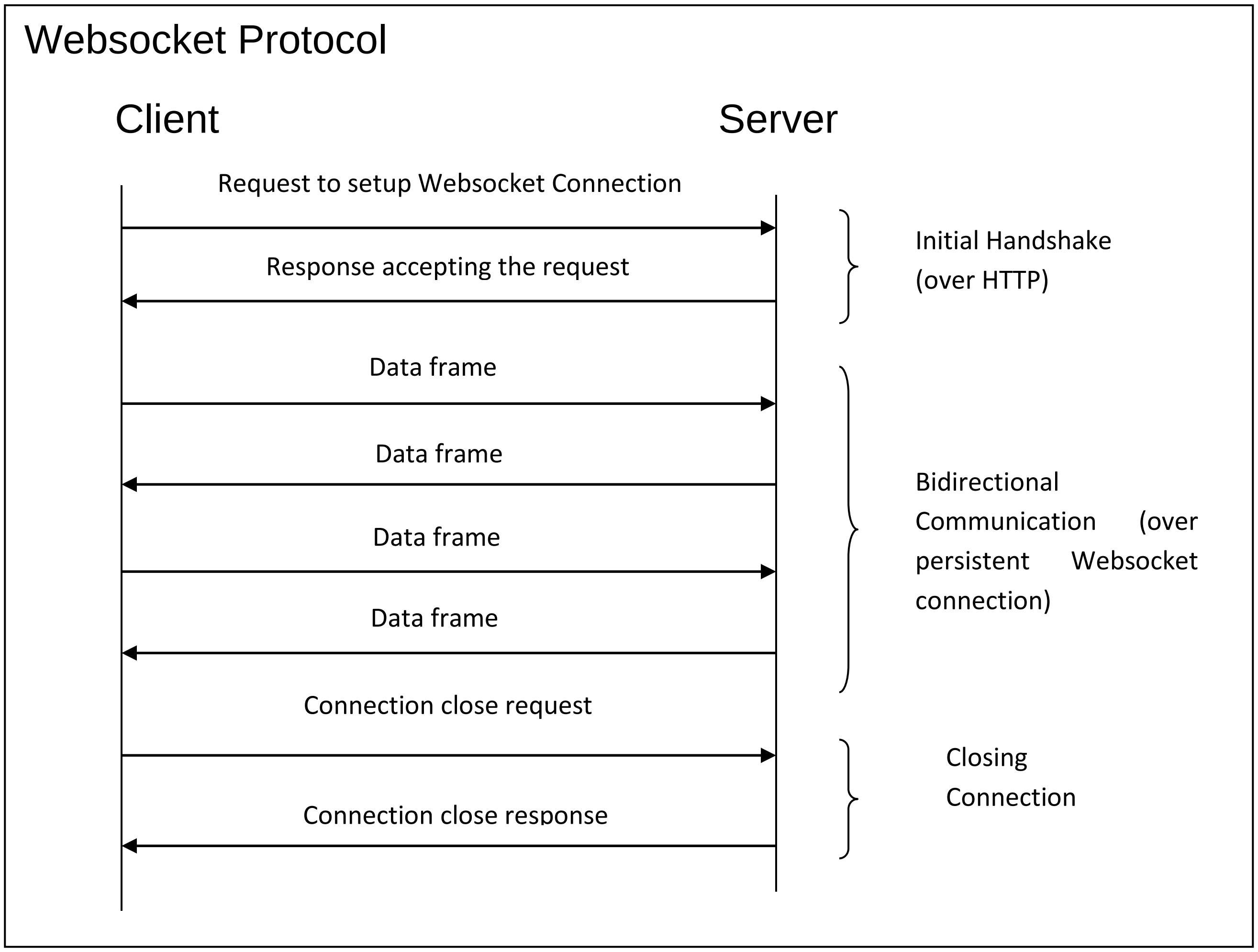

Websocket Protocol
Client
Server
Request to setup Websocket Connection
Response accepting the request
Initial Handshake (over HTTP)
Data frame
Data frame
Data frame
Data frame
Bidirectional Communication (over persistent Websocket connection)
Connection close request
Connection close response
Closing Connection

Figure 4.1.5: Exclusive Pair model used by Websocket APIs.

1.4 IoT Enabling Technologies:

IOT is enabled by several technologies including wireless sensor networks, cloud computing, Big Data analytics, embedded systems, security protocols and architectures, communication protocols, web services, mobile Internet and semantic search engines. This section provides an overview of some of these technologies which play a key role in IoT

1.4.1 Wireless sensor networks:

A wireless Sensor Network (WSN) comprises of distributed devices with Sensors which are used to monitor the environmental and physical conditions. a WSN consist of a number of end-nodes and routers and a coordinator. The coordinator collects the data from all the nodes. Coordinator also acts as a gateway that connects the WSN to the Internet. Some examples are WSNs used in IoT systems are described as follows:

- Weather monitoring systems use WSNs in which the nodes collect temperature, humidity and other data, which is aggregated and analyzed
- Indoor air quality monitoring systems use WSNs to collect data on the indoor air quality and concentration of various gaseous
- Soil moisture monitoring systems use WSNs to monitor soil moisture at various locations
- Surveillance systems use WSNs for collecting surveillance data(such as motion detection data)
- Smart grids use WSNs for monitoring the grid at various point
- Structural health monitoring systems use WSNs to monitor the health of structures (buildings, bridges)by collecting vibration data from sensor nodes deployed at various points at the structure

WSNs are enabled by wireless communication protocols such as IEEE 802.15.4.ZigBee is one of the most popular wireless technologies used by WSNs. ZigBee specifications are based on IEE 802.15.4.ZigBee operates at 2.4GHZ frequency and offer data rates up to 250KB/s and range from 10 to 100 meters depending on the power output and environmental conditions. The power of WSNs lies in their ability to deploy large number of low cost and low power-sensing nodes for continuous monitoring of environmental and physical conditions .

WSNs are self organizing networks. Since WSNs have large number of nodes, manual configuration of each node is not possible. The self organizing capability of WSN makes the network robust. In the event of failure of some node or addition of new nodes to the network, the network can reconfigure itself

1.4.2 Cloud Computing:

Cloud computing is transformative computing paradigm that involves delivering applications and services over the Internet. Cloud computing involves provisioning of computing, networking and storage resources on demand and providing these resources as metered services to the users, in a “Pay As You Go” model. Cloud computing assets can be provisioned on request by the clients, without requiring communications with the cloud specialist organization. The way toward provisioning assets is robotized. Distributed computing assets can be gotten to over the system utilizing standard get to instruments that give platform=Independent access using heterogeneous customer stages, for example, work stations Laptops tablets and Smart telephones. The registering and capacity assets given by cloud specialist organizations are pooled to serve numerous clients utilizing multi-occupancy. Multi-occupant parts of the cloud enable numerous clients to be served by the same physical equipment. Clients are allocated virtual resources that run on top of the physical resources.

Cloud computing services are offered to users in different forms.

- **Infrastructure as a service (IASS):** IASS provides the users the ability to provision computing and storage resources .these resources are provided to the users as virtual machine instances and virtual storage. Users can start, stop configure and manage the virtual machine instances and virtual storage. Users can deploy operating systems and applications of their choice on the virtual resources provisioned in the cloud. The cloud service provider manages the underlined infrastructure. Virtual resources provisioned by the users are build based on pay-per-use paradigm.
- **Platform as a service(PAAS):** PASS provides the users the ability to develop and deploy applications in cloud using the development tools, application programming interfaces (API’s), software libraries and services provided by the cloud service including servers networks operating systems and storage/ the users themselves are responsible for developing, deploying, configuring, managing applications on the cloud infrastructure.
- **Software as a Service (SAAS):** SAAS provides the users a complete software application or the user interface to the application itself. The cloud service provider manages the underlying cloud infrastructure including servers, networks, operating systems, storage and application software, and the user is unaware of underlying architecture of the cloud.

Applications are provided to the user through a client interface. SAAS applications are platform independent and can be accessed by various client devices such as workstations, laptops, tablets and Smart phones, running different operating system. since the cloud service provider manages both the application and data, the users are able to access the applications from anywhere.

Big Data Analytics:

Big Data is defined as collection of data whose volume, velocity, variety is so large that it is difficult to store, manage, process and analyze the data using traditional database and data processing tools. Big data analytics involves several steps starting from data cleansing, data managing, data processing and visualization. Some examples of Big data generated by Iot system are described as follows:

- Sensor data generated by Iot system such as weather monitoring system
- Machine sensor data collected from sensors embedded in industrial and energy systems for monitoring their health and detecting failures
- Health and fitness data generated by IoT device such as wearable fitness bands.
- Data generated by retail inventory monitory monitoring systems.

The underlying characteristics of big Data include:

- **Volume:** Though there is no fixed threshold for the volume of data to be considered as a big data, however, typically, the term big data is used for massive scale data that is difficult to store, manage and process using traditional database and data processing architectures. The volume of data generated by modern IT, industrial and healthcare systems, for example, is growing exponentially driven by the lowering costs of data storage and processing architectures and the need to extract valuable insights from the data to improve business processes, efficiency and service to consumers.
- **Velocity:** velocity is another important characteristic of Big data analytics and the primary reason for exponential growth of data. Velocity of data refers to hoe fast the data is generated and how frequently it varies. Modern IT, Industrial and other systems are generating data at increasingly higher speeds.

- **Variety:** variety refers to the forms of the data. Big data comes in different forms such as structured or unstructured data, including text data, image, audio, video and sensor data.

1.4.5 Communication Protocols:

Communication protocols form the backbone of IoT systems and enable network connectivity and coupling to applications. Communication protocols allow devices to exchange data over the network. These protocols define the data exchange formats, data encoding, addressing schemes for devices and routing of packets from source to destination. Other functions of the protocols include sequence control, flow control and retransmission of lost packets.

1.4.6 Embedded Systems:

An embedded system is a computer system that has computer hardware and software embedded to perform specific tasks. In contrast to general purpose computers or personal computers which can perform various tasks, embedded systems are designed to perform a specific set of tasks. Key component of an embedded system include microprocessor, micro controller , memory, networking unit, input/output unit and storage. Some embedded systems have specialized processors such as digital signal processors, graphics processors and application specific processors. Embedded systems rum embedded running operating systems such as real time operating systems(RTOS). Embedded systems range from low cost miniaturized devices such as digital watches to devices such as digital cameras, point of sale terminals, vending machines, appliances, etc.

1.5 IoT Levels & Deploying Templates:

An IoT system comprises of following components:

- **Device:** An IoT device allows identification, remote sensing, actuating and remote monitoring capabilities

- **Resources:** resources re software components on IoT devices for accessing, processing and storing sensor information, or controlling actuators connected to the data device. Resources also include the software components that enable network access for the device
- **Control Service:** it is a native service that runs on the device and interacts with the web services. Controller service sends data from the device to the web service and receives commands from the application(via web services) for controlling device.
- **Web Service:** web services serve as a link between an IoT device, application, database and analysis components. Web services can be either implemented using HTTP and REST principles or using WebSocket Protocol.

Comparison of REST and WebSocket:

- ✓ **Stateless/stateful:** REST services are stateless in nature, each request contains all the information needed to process it. Requests are independent of each other. WebSocket on the other hand is stateful in nature where the server maintains the state and is aware of all the open connections
- ✓ **Uni-directional/Bi-directional:** Rest services operate over HTTP and are uni-directional. Request ois always sent by a client and the server responds to the requests. On the other hand WebSocket is a bi-directional protocol and allows both client and server to send messages to each other.
- ✓ **TCP connections:** For REST services each HTTP requests involves setting up a new TCP connection. WebSocket on the other hand involves a single TCP connection over which the client and server communicate in full duplex mode.
- ✓ **Header Overheader:** Rest services operate over HTTP, and each request is independent of others. Thus each request carries HTTP headers which are an Overhead. Due the overhead of HTTP headers, REST is not suitable for real-time applications .WebSockets on the other hand does not involve overhead of header. After the initial Handshake (that happens over HTTP), the client and server exchange messages with minimal frame information. Thus WebSocket is suitable for real time applications.

- ✓ **Scalability:** scalability is easier in the case of REST services as requests ate independent and no state information needs to be maintained by the server. Thus both horizontal (scaling out) and vertical scaling (scaling up) solutions are possible for REST services. For WebSocktes, horizontal scaling can be cumbersome due to the stateful nature of the communication. Since the server maintains the state of connection, vertical scaling is easier for WebSockets than horizontal scaling

Analysis component:

The analysis component is responsible for analyzing the IoT data and generates results in a form which are easy for the user to understand. Analysis of IoT data can be performed either locally or in the cloud. Analyzed results are stored in the local or cloud databases

1.6 Application:

IoT application provides an interface that the users can use to control and monitor various aspects of the IoT system. Application also allows users to view the system status and view the processed data.

IoT level 1:

A level 1 IoT system has a single node or device that performs sensing and /actuation, stores data performs analysis and hosts the application as shown In figure 1.6.1. Level 1 IoT systems are suitable for modeling low cost and low complexity solutions where the data involved is not big and the analysis requirements are not computationally intensive.

IoT level 2:

A level-2 IoT system has a single node that performs sensing and or actuation and local analysis is shown in figure 1.2 data is stored in the cloud and application is usually cloud based. Level -2 systems are suitable for solutions where the data involved is big, however, the primary analysis requirement is not computationally intensive and can be done locally itself.

IoT level 3:

A level-3 IoT system has a single node. Data is stored and analyzed in the cloud and application is cloud based as shown in figure 1.6.3. level-3IoT systems are suitable for solutions where the data involved is big and the analysis requirements are computationally intensive.

IoT level 4:

A Level-4 IoT system has multiple nodes that perform local analysis. data is stored I the cloud and application is cloud based as shown in fig 1.6.4. Level-4 contains local and cloud based observer nodes which can subscribe to and receive information collected in the clod from IoT device. Observer nodes can process information and use it for various applications. However, observer does not perform any control functions. These systems are suitable for solutions where multiple nodes are required, the data involved is big and the analysis requirements are computationally intensive.

IoT level 5:

A Level-5 IoT system has multiple end nodes and one coordinator as shown I figure 1.6.5. The end nodes that perform sensing or actuation. Coordinator node collects the data from the end nodes and sends to the cloud. Data is analyses and stored in the cloud and application is cloud based. These systems are suitable for solutions based on wireless sensor networks, in which the data involved is big and the analysis requirements are computationally intensive

IoT level 6:

A Level-6 IoT system has multiple independent end nodes that performs sensing and actuation and sends the data to cloud. Data is stored in the cloud and application is cloud based as shown in fig 1.6.6. the analytics component analyzes the data and stores the results in the clod database. The results are visualized with the cloud based application. The centralized controller is aware of the status of all the end nodes and sends control commands to the nodes.

4.2 Raspberry Pi

Getting Started with Raspberry pi

The Raspberry Pi is a dedit card-sized computer that plugs into your PC and a keyboard. It is a capable little computer which can be used in electronics projects, and for many of the things that your desktop PC does, like spreadsheets, word processing, aurfing internet, and playing games. It also plays high-definition video.

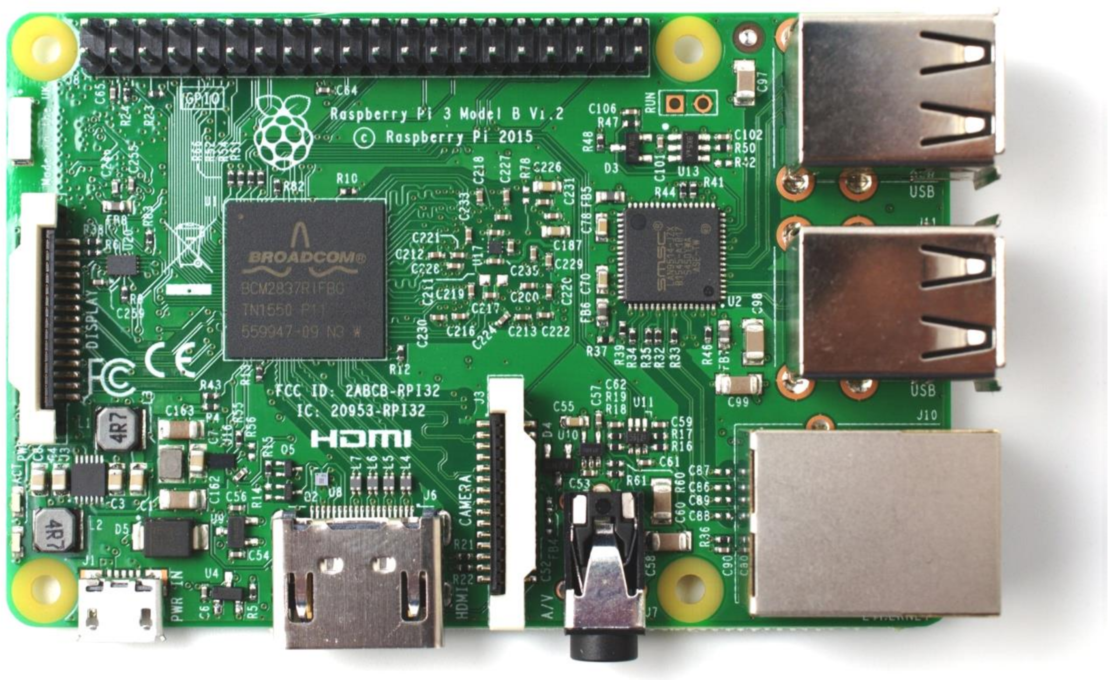
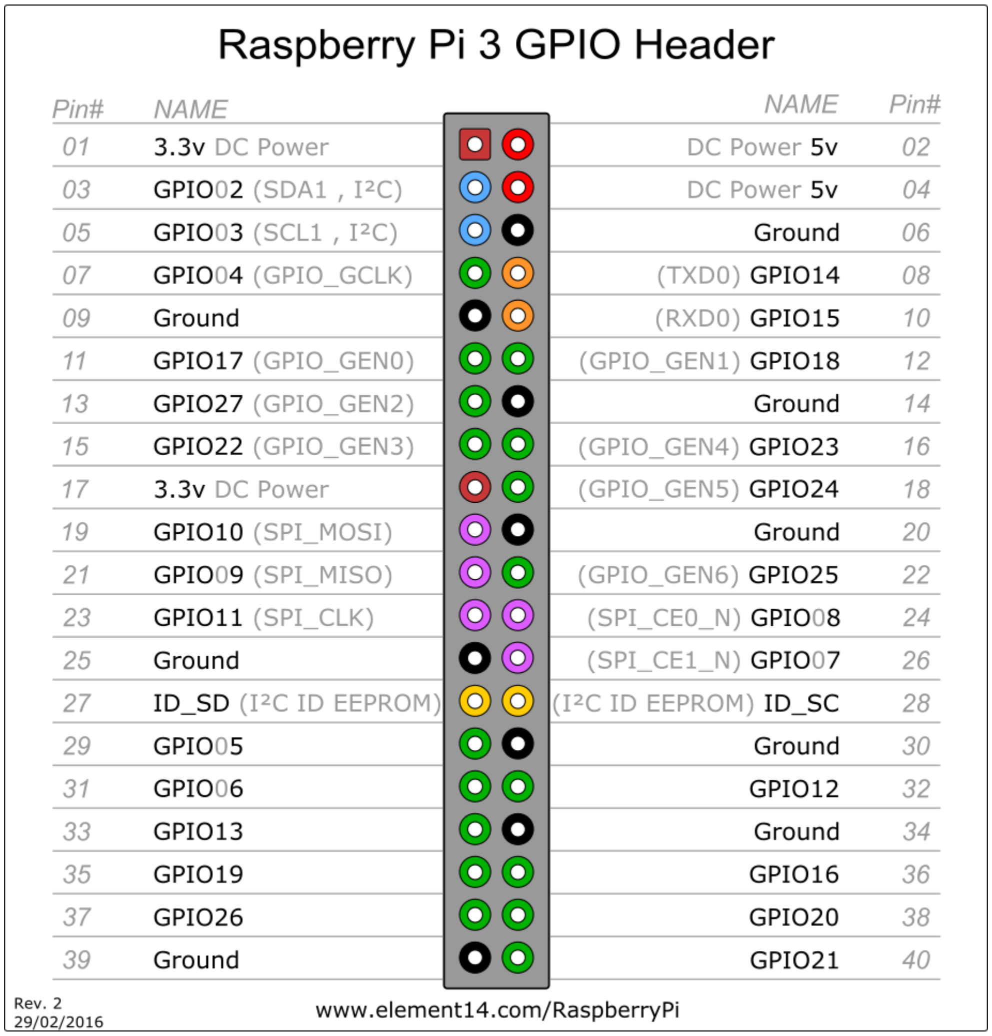


Figure 4.2.1: RaspberryPi 2 Hardware.

Rapberry pi Pin layout:



Raspberry Pi 3 GPIO Header

Pin#	NAME			NAME	Pin#
01	3.3v DC Power	🟥	🔴	DC Power 5v	02
03	GPIO02 (SDA1 , I²C)	🔵	🔴	DC Power 5v	04
05	GPIO03 (SCL1 , I²C)	🔵	⚫	Ground	06
07	GPIO04 (GPIO_GCLK)	🟢	🟠	(TXD0) GPIO14	08
09	Ground	⚫	🟠	(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)	🟢	🟢	(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)	🟢	⚫	Ground	14
15	GPIO22 (GPIO_GEN3)	🟢	🟢	(GPIO_GEN4) GPIO23	16
17	3.3v DC Power	🔴	🟢	(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)	🟣	⚫	Ground	20
21	GPIO09 (SPI_MISO)	🟣	🟢	(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)	🟣	🟣	(SPI_CE0_N) GPIO08	24
25	Ground	⚫	🟣	(SPI_CE1_N) GPIO07	26
27	ID_SD (I²C ID EEPROM)	🟡	🟡	(I²C ID EEPROM) ID_SC	28
29	GPIO05	🟢	⚫	Ground	30
31	GPIO06	🟢	🟢	GPIO12	32
33	GPIO13	🟢	⚫	Ground	34
35	GPIO19	🟢	🟢	GPIO16	36
37	GPIO26	🟢	🟢	GPIO20	38
39	Ground	⚫	🟢	GPIO21	40

Rev. 2
29/02/2016

www.element14.com/RaspberryPi

Figure 4.2.2:RaspberryPi configuration pin.

Setup Raspberry Pi:

While the Raspberry Pi is a mini size computer, we still need to install software on it to carry out operations such as to gather data, store it and display it. The software is based on the Linux Operating System . On the off chance that you don't have any recognition with Linux, then you are going to have an introduction to it.. On the off chance that you are more certain and know your sudo from your system cover, there is a Raspberry Pi Quick Set-up segment for the product stacking process in the indices.

One of the imperative parts of the diverse activities that we will deal with is that they have a typical base. The procedure in the accompanying segments will shape that base, so that with each new estimation method we bring with an alternate sensor, we will expect that our Raspberry Pi condition has been set up with the accompanying programming. This will permit each venture/section to be drawn closer independently if coveted.

5.0 Processing Humidity Data at Raspberry pi

In this experiment DHT-11 sensor data (Temperature, Humidity) will be processes by raspberry pi and it maintains the local storage of data. Raspberry as an edge device calculates the dew point from temperature and humidity, calculated dew point will be sent to the cloud for further analytics.

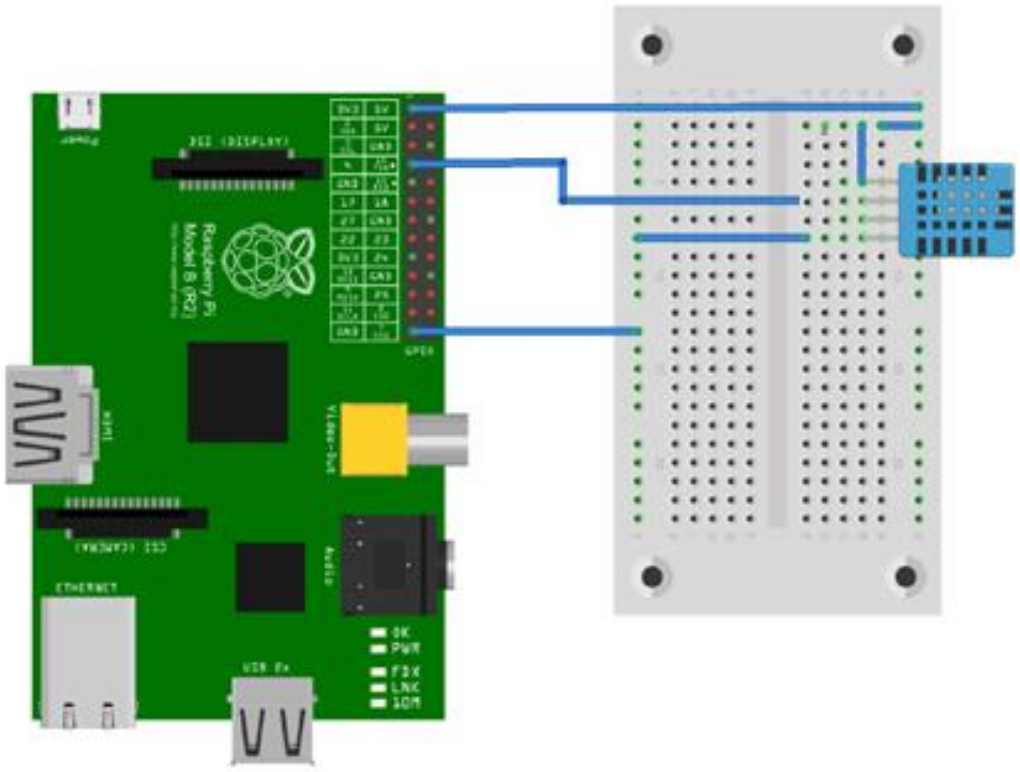


Figure 4.2.3: Connect DHT-11 sensor with raspberry pi as shown

4.3 Python

Python is an interpreted, object-oriented, high-level programming language ,with dynamic semantics. Its abnormal state worked in information structures, joined with dynamic writing and dynamic official, make it extremely appealing for Rapid Application Development, and in addition for use as a scripting or paste dialect to associate existing parts together. Python's straightforward, simple to learn grammar underlines intelligibility and thusly diminishes the cost of program support. Python underpins modules and bundles, which supports program particularity and code reuse. The Python translator and the broad standard library are accessible in source or paired frame without charge for every single real stage, and can be unreservedly appropriated.

Frequently, software engineers begin to look all starry eyed at Python as a result of the expanded efficiency it gives. Since there is no aggregation step, the alter test-investigate cycle is unfathomably quick. Investigating Python projects is simple: a bug or awful info will never bring about a division blame. Rather, when the mediator finds a mistake, it raises a special case. At the point when the program doesn't get the exemption, the mediator prints a stack follow. A source level debugger permits investigation of neighborhood and worldwide factors, assessment of discretionary expressions, setting breakpoints, venturing through the code a line at any given moment, et cetera. The debugger is composed in Python itself, vouching for Python's reflective power. Then again, frequently the speediest approach to investigate a program is to add a couple print articulations to the source: the quick alter test-troubleshoot cycle makes this straightforward approach extremely compelling.

5. CODING AND TESTING

Objectdetection.py

```
################################################################################
# import the necessary packages
import argparse
from imutils.object_detection import non_max_suppression
import imutils
from imutils import paths
import numpy as np
import cv2
import time
import pyttsx

################################################################################
#

cascade_src = 'bananahaar.xml'
video_src = 'video2.avi'
count = 0

#cap = cv2.VideoCapture(video_src) #Comment for real video capture
cap = cv2.VideoCapture('testimage2.jpg')
#cap = cv2.VideoCapture(0) #Uncomment for real time video
detector = cv2.CascadeClassifier(cascade_src) # Select the trained cascade classifier

################################################################################
##
```

```
while True:
    pedCount=0
    obCount= 0
    ret, img = cap.read()
    origImg = img

    if (type(img) == type(None)):
        break

    detector_cascade = cv2.CascadeClassifier(cascade_src) # Select the trained cascade classifier
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) # Converts the image into grayScale
image
    objects = detector_cascade.detectMultiScale(gray, scaleFactor=1.3,
            minNeighbors=20, minSize=(150, 150)) # Detect the number of cars in the video



    # loop over the cat faces and draw a rectangle surrounding each
    for (i, (x, y, w, h)) in enumerate(objects):
        cv2.rectangle(img, (x, y), (x + w, y + h), (0, 0, 255), 2)
        cv2.putText(img, "Object #{}".format(i + 1), (x, y - 10),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.55, (0, 0, 255), 2)
        obCount= obCount+1

    cv2.imshow('Object_Recognized', img) # Display the frames

    if obCount >= 1:
        engine = pyttsx.init()
        Voice = "There are" +str(obCount)+ "Bananas"
        engine.say(Voice)
        engine.runAndWait()
```

```
####################################################################################
###################
    # initializing HOG descriptor (or person detector)
    hog = cv2.HOGDescriptor()
    hog.setSVMDetector(cv2.HOGDescriptor_getDefaultPeopleDetector())
    image = imutils.resize(origImg, width=min(400, origImg.shape[1]))
    # detect people in the image
    (rects, weights) = hog.detectMultiScale(image, winStride=(4, 4),
            padding=(8, 8), scale=1.05)
    # draw the original bounding boxes
    for (x, y, w, h) in rects:
            cv2.rectangle(image, (x, y), (x + w, y + h), (0, 0, 255), 2)

    # apply non-maxima suppression to the bounding boxes using a
    # fairly large overlap threshold to try to maintain overlapping
    # boxes that are still people
    rects = np.array([[x, y, x + w, y + h] for (x, y, w, h) in rects])
    pick = non_max_suppression(rects, probs=None, overlapThresh=0.65)

    # draw the final bounding boxes
    for (xA, yA, xB, yB) in pick:
            cv2.rectangle(image, (xA, yA), (xB, yB), (0, 255, 0), 2)
            pedCount= pedCount+1
    # show the output images
    cv2.imshow("Banana", image)

    if pedCount >= 1:
        engine = pyttsx.init()
        #engine.say('One Pedestrian in Front')
        Voice = "and There are" +str(pedCount)+ "pedestrians in front"
        engine.say(Voice)
        engine.runAndWait()


    if cv2.waitKey(33) == 27: #press 'ESC' key to stop the video
        break
    time.sleep(3)

cv2.destroyAllWindows()
```

Cardetection.py

```
################################################################################
# import the necessary packages
from imutils.object_detectionimport non_max_suppression
from imutils importpaths
import numpy as np
import argparse
import imutils
import cv2
import time
import pyttsx

################################################################################
#

cascade_src = 'checkcas.xml'
#video_src = 'video2.avi'
count = 0

#cap = cv2.VideoCapture(video_src) #Comment for real video capture
```

```
cap = cv2.VideoCapture('testimage2.jpg')
#cap = cv2.VideoCapture(0) #Uncomment for real time video
detector = cv2.CascadeClassifier(cascade_src) # Select the trained cascade classifier

################################################################################
##

while True:
   pedCount=0
   obCount= 0
   ret, img = cap.read()
   origImg = img

   if (type(img) == type(None)):
      break

   detector_cascade = cv2.CascadeClassifier(cascade_src) # Select the trained cascade classifier
   gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) # Converts the image into grayScale
image
   objects = detector_cascade.detectMultiScale(gray, scaleFactor=1.3,
            minNeighbors=20, minSize=(150, 150)) # Detect the number of cars in the video



   # loop over the cat faces and draw a rectangle surrounding each
   for (i, (x, y, w, h)) in enumerate(objects):
         cv2.rectangle(img, (x, y), (x + w, y + h), (0, 0, 255), 2)
         cv2.putText(img, "Object #{}".format(i + 1), (x, y - 10),
                        cv2.FONT_HERSHEY_SIMPLEX, 0.55, (0, 0, 255), 2)
         obCount= obCount+1

   cv2.imshow('Object_Recognized', img) # Display the frames
```

```
    if obCount >= 1:
        engine = pyttsx.init()
        Voice = "There are" +str(obCount)+ "Cars"
        engine.say(Voice)
        engine.runAndWait()


##################################################################################
##################
    #initializeHOG descriptor (person detector)
    hog = cv2.HOGDescriptor()
    hog.setSVMDetector(cv2.HOGDescriptor_getDefaultPeopleDetector())
    image = imutils.resize(origImg, width=min(400, origImg.shape[1]))
    # detect people in the image
    (rects, weights) = hog.detectMultiScale(image, winStride=(4, 4),
                padding=(8, 8), scale=1.05)
    # draw the original bounding boxes
    for (x, y, w, h) in rects:
                cv2.rectangle(image, (x, y), (x + w, y + h), (0, 0, 255), 2)

    # apply non-maxima suppression to the bounding boxes using a
    # fairly large overlap threshold to try to maintain overlapping
    # boxes that are still people
    rects = np.array([[x, y, x + w, y + h] for (x, y, w, h) in rects])
    pick = non_max_suppression(rects, probs=None, overlapThresh=0.65)

    # draw the final bounding boxes
    for (xA, yA, xB, yB) in pick:
                cv2.rectangle(image, (xA, yA), (xB, yB), (0, 255, 0), 2)
                pedCount= pedCount+1
    # show the output images
    cv2.imshow("car ", image)

    if pedCount >= 1:
        engine = pyttsx.init()
        #engine.say('One Pedestrian in Front')
        Voice = "There are" +str(pedCount)+ "pedestrian in front"
        engine.say(Voice)
        engine.runAndWait()



    if cv2.waitKey(33) == 27: #press 'ESC' key to stop the video
        break
    time.sleep(3)

cv2.destroyAllWindows()
```

pedisterian.py

```
##########################################################################################
##################
# importthe necessarypackages
from imutils.object_detection import non_max_suppression
import numpy as np
import argparse
import cv2
import imutils
from imutils import paths
from __future__ import print_function

### construct the argument parse and parse the arguments
##ap = argparse.ArgumentParser()
```

```
##ap.add_argument("-i", "--testimage2.jpg", required=True, help="/home/pi/Code/SmartCap")
##args = vars(ap.parse_args())

# initialize the HOG descriptor/person detector
hog = cv2.HOGDescriptor()
hog.setSVMDetector(cv2.HOGDescriptor_getDefaultPeopleDetector())

# loop over the image paths
for imagePath in paths.list_images(args["testimage2.jpg"]):
        # load the image and resize it to (1) reduce detection time
        # and (2) improve detection accuracy
        image = cv2.imread('download1.jpg')
        image = imutils.resize(image, width=min(400, image.shape[1]))
        orig = image.copy()

        # detect people in the image
        (rects, weights) = hog.detectMultiScale(image, winStride=(4, 4),
                padding=(8, 8), scale=1.05)

        # draw the original bounding boxes
        for (x, y, w, h) in rects:
                cv2.rectangle(orig, (x, y), (x + w, y + h), (0, 0, 255), 2)

        # apply non-maxima suppression to the bounding boxes using a
        # fairly large overlap threshold to try to maintain overlapping
        # boxes that are still people
        rects = np.array([[x, y, x + w, y + h] for (x, y, w, h) in rects])
        pick = non_max_suppression(rects, probs=None, overlapThresh=0.65)

        # draw the final bounding boxes
        for (xA, yA, xB, yB) in pick:
                cv2.rectangle(image, (xA, yA), (xB, yB), (0, 255, 0), 2)

        # show some information on the number of bounding boxes
        filename = imagePath[imagePath.rfind("/") + 1:]
        print("[INFO] {}: {} original boxes, {} after suppression".format(
                filename, len(rects), len(pick)))

        # show the output images
        cv2.imshow("Before NMS", orig)
        cv2.imshow("After NMS", image)
        cv2.waitKey(0)
```

5.2 TESTING

5.2.1 SYSTEM TESTING

The porpose of testing is to detect errors. Testing is the process of trying to discover every anticipated fault or bug in a work product. It enhances a way to check the functionality of components, byproducts , assemblies and/or a finished product It is the process of testing software with the best intent of assuring that the Software system meets its requirements and user expectations and does not fail in an unsatisfied manner. There are various types of test. Each test type addresses a unique testing requirement.

TYPES OF TESTS

1. Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All choice branches and interior code stream ought to be approved. It is the trying of individual programming units of the application .it is done after the fulfillment of an individual unit before coordination. This is an auxiliary testing, that depends on learning of its development and is obtrusive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

2. Integration testing

Integration tests are designed to test incorporated programming parts to decide whether they really keep running as one program. Testing is occasion driven and is more worried with the fundamental result of screens or fields. Coordination tests exhibit that in spite of the fact that the segments were exclusively fulfillment, as appeared by effectively unit testing, the mix of parts is right and steady. Joining testing is particularly gone for uncovering the issues that emerge from the blend of segments.

3. Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

- Valid Input : identified classes of valid input must be accepted.
- Invalid Input : identified classes of invalid input must be rejected.
- Functions : identified functions must be exercised.
- Output : identified classes of application outputs must be exercised.
- Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. Moreover, precise scope relating to distinguish Business prepare streams; information fields, predefined forms, and progressive procedures must be considered for testing. Before useful testing is finished, extra tests are recognized and the successful estimation of current tests is resolved.

4. System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

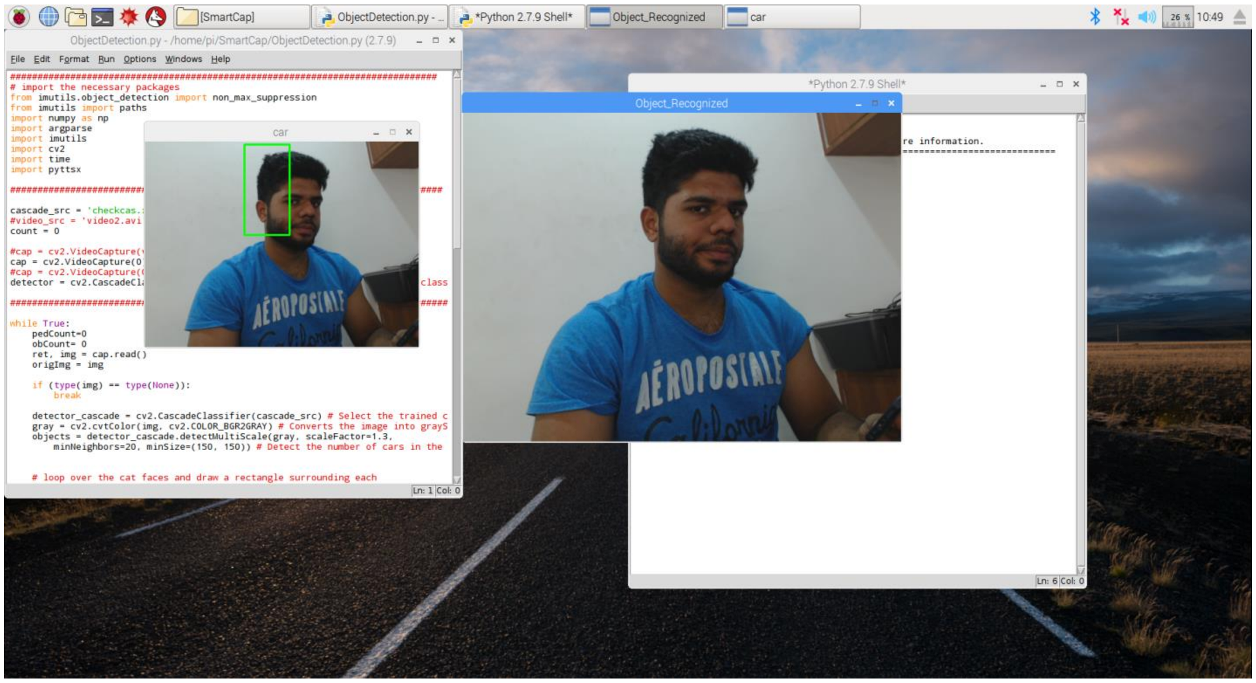
5. White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

6. Black Box Testing

Black Box Testing is testing the product with no information of the inward workings, structure or dialect of the module being tried. Discovery tests, as most different sorts of tests, must be composed from a complete source archive, for example, particular or prerequisites report, for example, detail or necessities record. It is a trying in which the product under test is dealt with, as a discovery .you can't "see" into it. The test gives data sources and reacts to yields without considering how the product functions.

5.2.4 Testing the Output


[SmartCap]
ObjectDetection.py - ...
Python 2.7.9 Shell
Object_Recognized
car
10:49
ObjectDetection.py - /home/pi/SmartCap/ObjectDetection.py (2.7.9)
File Edit Format Run Options Windows Help
Ln: 1 Col: 0
Ln: 6 Col: 0

```
###############################################################################
# import the necessary packages
from imutils.object_detection import non_max_suppression
from imutils import paths
import numpy as np
import argparse
import imutils
import cv2
import time
import pyttsx

##########################                                                ####

cascade_src = 'checkcas.
#video_src = 'video2.avi
count = 0

#cap = cv2.VideoCapture(
cap = cv2.VideoCapture(0
#cap = cv2.VideoCapture(
detector = cv2.CascadeCl                                                  class

##########################                                                #####

while True:
    pedCount=0
    obCount= 0
    ret, img = cap.read()
    origImg = img

    if (type(img) == type(None)):
        break

    detector_cascade = cv2.CascadeClassifier(cascade_src) # Select the trained c
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) # Converts the image into grayS
    objects = detector_cascade.detectMultiScale(gray, scaleFactor=1.3,
        minNeighbors=20, minSize=(150, 150)) # Detect the number of cars in the


    # loop over the cat faces and draw a rectangle surrounding each
```

```
re information.
==============================
```

Figure 5.2.1.: Sample webcam output

The program is tested by testing the output . If the output generated has green rectangular box over it indicates that the image is identified. If no box is generated then it signifies that the program is executed but the image is not identified.

5.2.5 Installation and testing of Raspberry Pi

NOOBS (New Out Of Box Software) is an easy operating system install manager for the Raspberry Pi to install Rasbian.

- REQUIRED
    - ➢ SD Card
    - ➢ Display and connectivity cables
    - ➢ Keyboard and mouse
    - ➢ Power supply
    - ➢ NOT ESSENTIAL BUT HELPFUL TO HAVE
    - ➢ Internet connection
    - ➢ Headphones

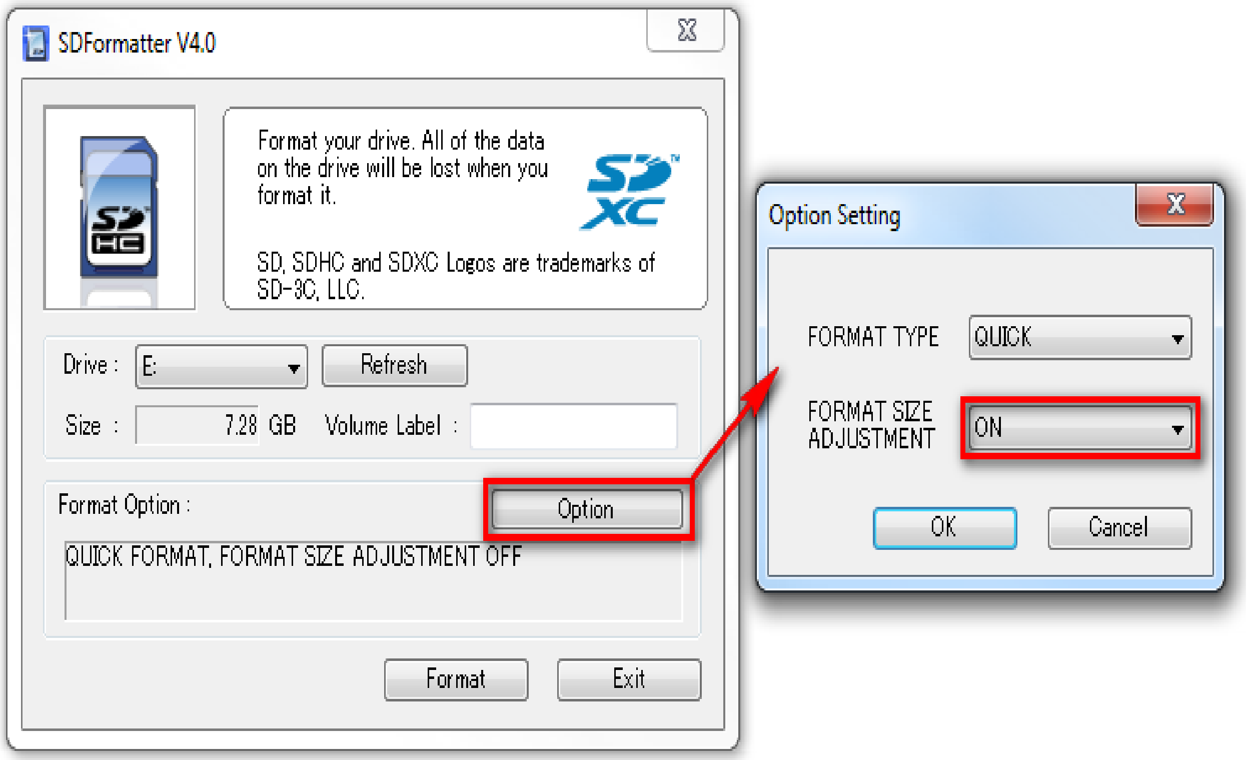
Prepare an SD Card

- ➢ Before you can start installing Raspbian, you'll have to get the installer set up on an SD card
- ➢ prepare your SD card by formatting it, and then installing the NOOBS software.

Format the Sd Card

- ➢ Download SD Formatter for Windows. Unzip it, and run Setup.exe. Follow along with the Install Shield Wizard to install SD Formatter.
- ➢ Open SD Formatter.
- ➢ Click Option and set FORMAT SIZE ADJUSTMENT to ON.

- Select your card from the Drive dropdown menu (if it wasn't selected automatically). Check and double-check that the drive letter is correct.
- Click Format, then click OK a couple times. You should get a Drive Format complete! pop up shortly.


SDFormatter V4.0
Format your drive. All of the data on the drive will be lost when you format it.
SD, SDHC and SDXC Logos are trademarks of SD-3C, LLC.
Refresh
Option
Format
Exit
Option Setting
OK
Cancel

Drive : E:
Size : 7.28 GB
Volume Label :
Format Option :
QUICK FORMAT, FORMAT SIZE ADJUSTMENT OFF

FORMAT TYPE: QUICK
FORMAT SIZE ADJUSTMENT: ON

Figure 5.2.5(1)L: formatting sd card

- Download NOOBS
  - Using a computer with an SD card reader, visit the Downloads page.
  - Click on the Download ZIP button under 'NOOBS (offline and network install)', and select a folder to save it to.
  - Extract the files from the zip
- Drag And Drop NOOBS FILES

- Once your SD card has been formatted, drag all the files in the extracted NOOBS folder and drop them onto the SD card drive.
- The necessary files will then be transferred to your SD card.
- When this process has finished, safely remove the SD card and insert it into your Raspberry Pi.

- Connect the display

There are two potential places to connect a display on the Pi:
either **HDMI** or **component video**.

- Connect USB Peripherals

Mouse and Key Board

- Insert SD Card and connect power cables

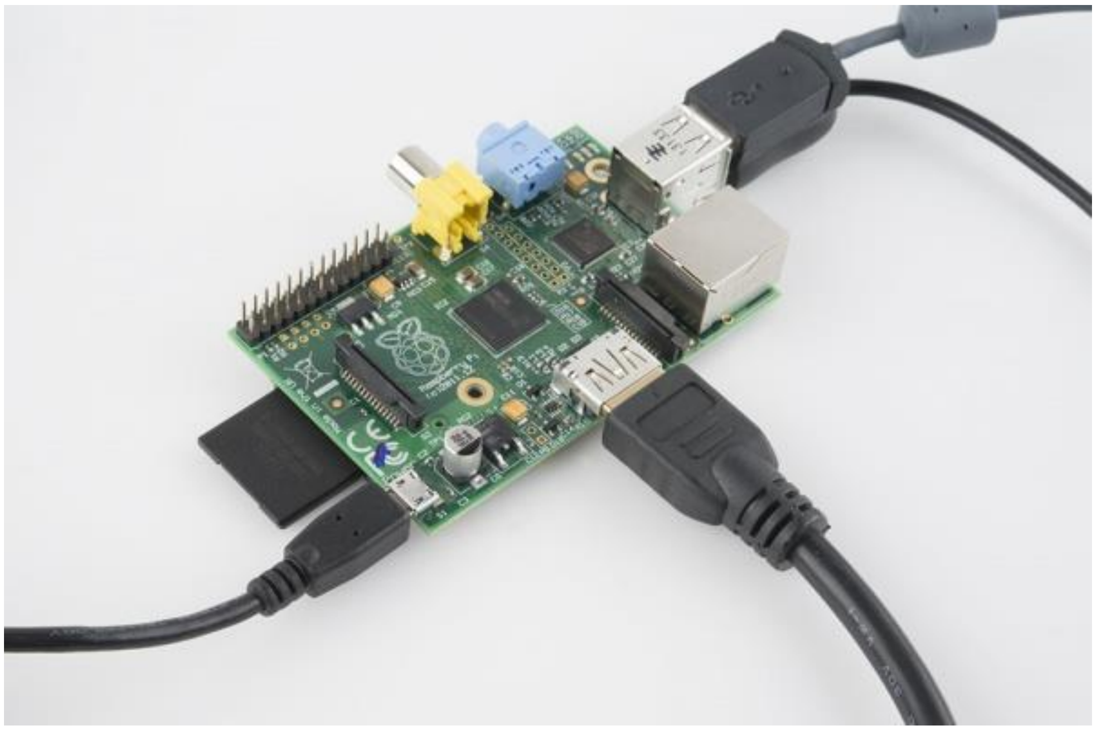
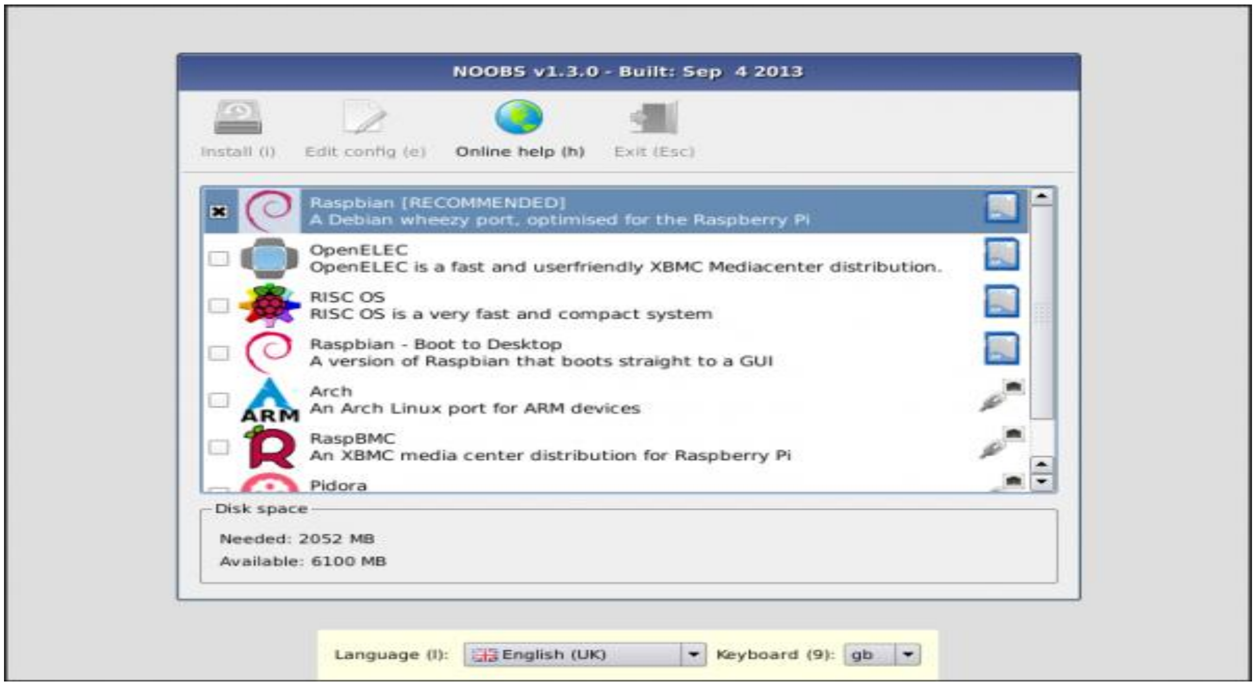


Figure 5.2.5(2): Peripherals wires connected to RaspberrtPi

- First Boot

  - ➢ Your Raspberry Pi will boot, and a window will appear with a list of different operating systems that you can install. We recommend that you use Raspbian – tick the box next to Raspbian and click on Install.
  - ➢ Raspbian will then run through its installation process. Note that this can take a while.
  - ➢ When the install process has completed, the Raspberry Pi configuration menu (raspi-config) will load. Here you are able to set the time and date for your region, enable a Raspberry Pi camera board, or even create users. You can exit this menu by using **Tab** on your keyboard to move to Finish.



[x] Raspbian [RECOMMENDED]
A Debian wheezy port, optimised for the Raspberry Pi

[ ] OpenELEC
OpenELEC is a fast and userfriendly XBMC Mediacenter distribution.

[ ] RISC OS
RISC OS is a very fast and compact system

[ ] Raspbian - Boot to Desktop
A version of Raspbian that boots straight to a GUI

[ ] Arch
An Arch Linux port for ARM devices

[ ] RaspBMC
An XBMC media center distribution for Raspberry Pi

Pidora

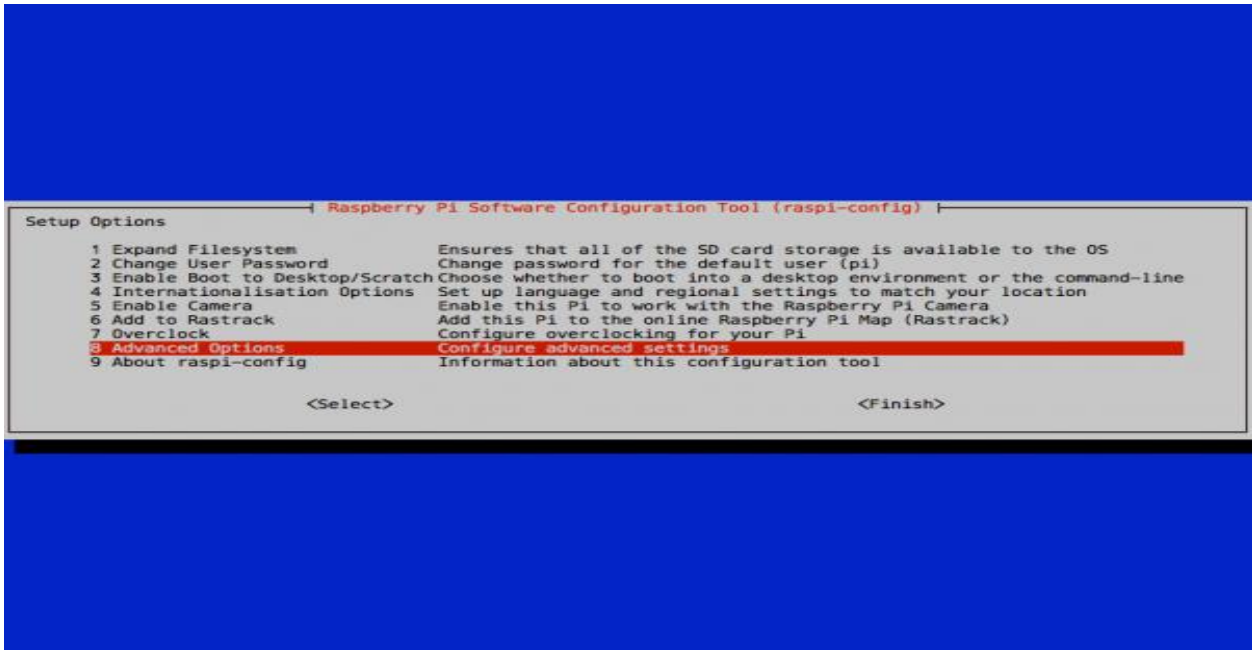
Needed: 2052 MB
Available: 6100 MB

Language (l): English (UK)   Keyboard (9): gb

Figure 5.2.5(2): Booting up

After selecting the image, you can modify the language setting. Then **click install** or press i to start the installation.

- Set Up Rasbian



```
                    ┤ Raspberry Pi Software Configuration Tool (raspi-config) ├
Setup Options

      1 Expand Filesystem              Ensures that all of the SD card storage is available to the OS
      2 Change User Password           Change password for the default user (pi)
      3 Enable Boot to Desktop/ScratchChoose whether to boot into a desktop environment or the command-line
      4 Internationalisation Options   Set up language and regional settings to match your location
      5 Enable Camera                  Enable this Pi to work with the Raspberry Pi Camera
      6 Add to Rastrack                Add this Pi to the online Raspberry Pi Map (Rastrack)
      7 Overclock                      Configure overclocking for your Pi
      8 Advanced Options               Configure advanced settings
      9 About raspi-config             Information about this configuration tool

                   <Select>                                             <Finish>
```

Figure 5.2.5(3): Expanding file system

- **Expand Filesystem** – Don't worry about this. NOOBS already did it for you.
- **Change User Password** – This step is recommended! Follow the on-screen directions to set a new password for you Pi. By default the password is set to *raspberry*, and the user is set to *pi*.
- **Enable Boot to Desktop/Scratch** – Select whether to boot into desktop or simply the text console. The console mode will obviously boot faster, and you can type Startx to open the GUI. Booting to desktop may be easier for those more comfortable with Windows or Mac, though.
- **Internationalization Options** – Here you can adjust the timezone, keyboard layout, and language of your Pi. These changes take a while to be made, so be patient.
- **Enable Camera** – If you've got a Raspberry Pi Camera, this is the setting for you.
- **Add to Rastrack** – If you want your Pi to be documented on Rastrack.

- Once you’ve made all of your adjustments, scroll down and over to <Finish> and allow the Pi to Reboot.
- If you booted into console mode, type Startx to open the GUI.
- Exploring Raspbian
- Raspbian comes with a variety of useful software tools

Raspberry Pi Libraries Installation

- Rpi.GPIO INSTALLATION
    - ➢ $ sudo apt-get update
    - ➢ $ sudo apt-get install python-dev
    - ➢ $ sudo apt-get install python-rpi.gpio

PIGPIO LIBRARY INSTALLATION

- pigpio is a library for the Raspberry which allows control of the General Purpose Input Outputs (GPIO).pigpio works on all versions of the Pi.
- Download and install
- wget abyz.co.uk/rpi/pigpio/pigpio.zip
- unzip pigpio.zip
- cd PIGPIO
- make
- sudo make install
- To start the pigpio daemon
- sudopigpiod

6. SYSTEM DESIGN

6.1 INPUT DESIGN

The info configuration is the connection between the data framework and the client. It includes the creating determination and methods for information arrangement and those means are important to put exchange information into a usable frame for handling can be accomplished by reviewing the PC to peruse information from a composed or printed record or it can happen by having individuals entering the information specifically into the framework. The plan of information concentrates on controlling the measure of information required, controlling the blunders, dodging delay, maintaining a strategic distance from additional means and keeping the procedure basic. The information is outlined in such a path in this way, to the point that it gives security and convenience with holding the protection.

Input Design considered the accompanying things:

- What information ought to be given as info?
- How the information ought to be organized or coded?
- The discourse to control the working staff in giving info.
- Methods for get ready info approvals and ventures to take after when blunder happen.

OBJECTIVES

1. Input Design is the way toward changing over a client arranged portrayal of the contribution to a PC based framework. This outline is critical to maintain a strategic distance from blunders in the information input process and demonstrate the right bearing to the administration for getting right data from the electronic framework.

2.It is accomplished by making easy to understand screens for the information passage to deal with huge volume of information. The objective of outlining information is to make information section less demanding and to be free from blunders. The information section screen is outlined such that every one of the information controls can be performed. It additionally gives record seeing offices.

3. At the point when the information is entered it will check for its legitimacy. Information can be entered with the assistance of screens. Proper messages are given as when required so that the client won't be in maize of moment. In this manner the goal of info configuration is to make an information design that is anything but difficult to take after

6.2 OUTPUT DESIGN

A quality yield is one, which meets the prerequisites of the end client and presents the data obviously. In any framework aftereffects of preparing are imparted to the clients and to other framework through yields. In yield plan it is resolved how the data is to be dislodged for quick need and furthermore the printed copy yield. It is the most vital and direct source data to the client. Effective and astute yield configuration enhances the framework's relationship to help client basic leadership.

1. Planning PC yield ought to continue in a sorted out, well thoroughly considered way; the correct yield must be created while guaranteeing that each yield component is composed so individuals will discover the framework can utilize effortlessly and adequately. At the point when investigation outline PC yield, they ought to Identify the particular yield that is expected to meet the prerequisites.

2. Select techniques for introducing data.

3. Make archive, report, or different arrangements that contain data created by the framework.

The yield type of a data framework ought to achieve at least one of the accompanying destinations.

- Convey data about past exercises, current status or projections of the
- Future.
- Signal imperative occasions, openings, issues, or notices.
- Trigger an activity.
- Confirm an activity.

6.3 SYSTEM ARCHITECTURE

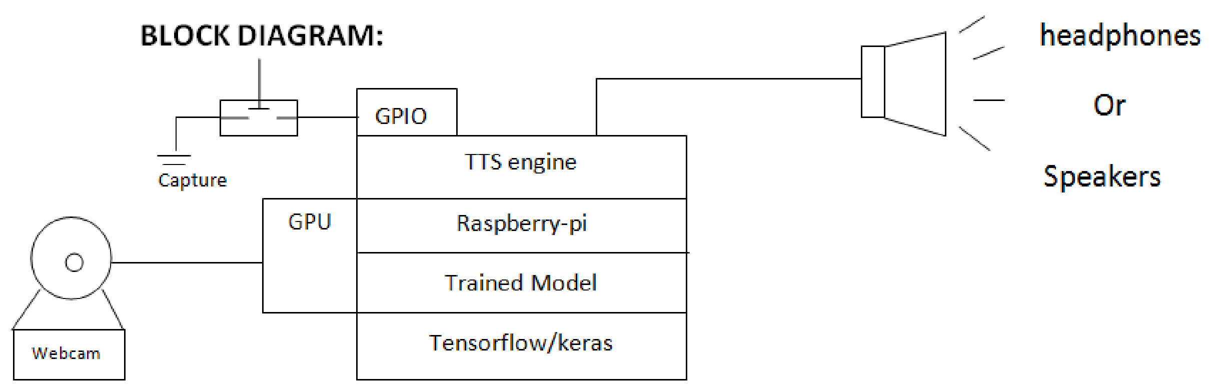


FIG 6.3.1 Architecture of the system

The above diagram shows the architecture of our project and the working of it. It shows how our system is designed and shows the flow among various elements throughout the system in an abstract view.

6.4 UML Concepts

The Unified Modelling Language (UML) is a standard language for writing software blue prints. The UML is a language for

- Visualizing
- Specifying

- Constructing
- Documenting the artefacts of a software intensive system.

The UML is a language which provides vocabulary and the rules for combining words in that vocabulary for the purpose of communication. A modelling language is a language whose vocabulary and the rules focus on the concesptual and physical representation of a system. Modelling yields an understanding of a system.

6.4.1Building Blocks of the UML

The vocabulary of the UML encompasses three kinds of building blocks:

- Things
- Relationships
- Diagrams

Things are the abstractions that are first-class citizens in a model; relationships tie these things together; diagrams group interesting collections of things.

1. Things in the UML

There are four kinds of things in the UML:

- Structural things
- Behavioral things
- Grouping things
- Annotational things

Structural things are the nouns of UML models. The structural things used in the project design are:

First, a **class** is a description of a set of objects that share the same attributes, operations, relationships and semantics.

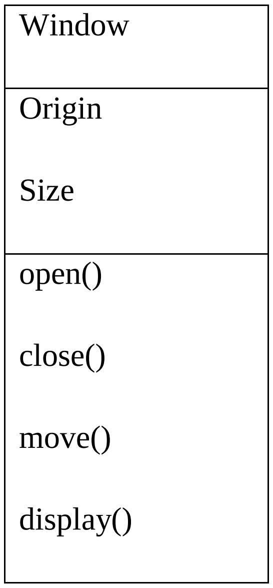


Fig: Classes

Second, a **use case** is a description of set of sequence of actions that a system performs that yields an observable result of value to particular actor.

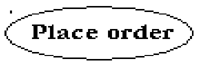


Fig: Use Cases

Third, a node is a physical element that exists at runtime and represents a computational resource, generally having at least some memory and often processing capability.

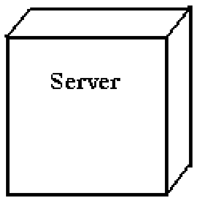


Fig: Nodes

Behavioral things are the dynamic parts of UML models. The behavioral thing used is:

Interaction:

An interaction is a behaviour that comprises a set of messages exchanged among a set of objects within a particular context to accomplish a specific purpose. An interaction involves a number of other elements, including messages, action sequences (the behaviour invoked by a message, and links (the connection between objects).

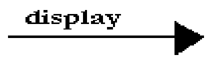


Fig: Messages

2. Relationships in the UML:

There are four kinds of relationships in the UML:

- Dependency
- Association
- Generalization
- Realization

A **dependency** is a semantic relationship between two things in which a change to one thing may affect the semantics of the other thing (the dependent thing).

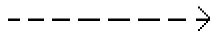


Fig: Dependencies

An **association** is a structural relationship that describes a set links, a link being a connection among objects. Aggregation is a special kind of association, representing a structural relationship between a whole and its parts.

Fig: Association

A **generalization** is a specialization/ generalization relationship in which objects of the specialized element (the child) are substitutable for objects of the generalized element (the parent).

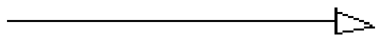


Fig: Generalization

A **realization** is a semantic relationship between classifiers, where in one classifier specifies a contract that another classifier guarantees to carry out.

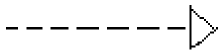


Fig: Realization

6.4.2 UML DIAGRAMS:

1. USE CASE DIAGRAM

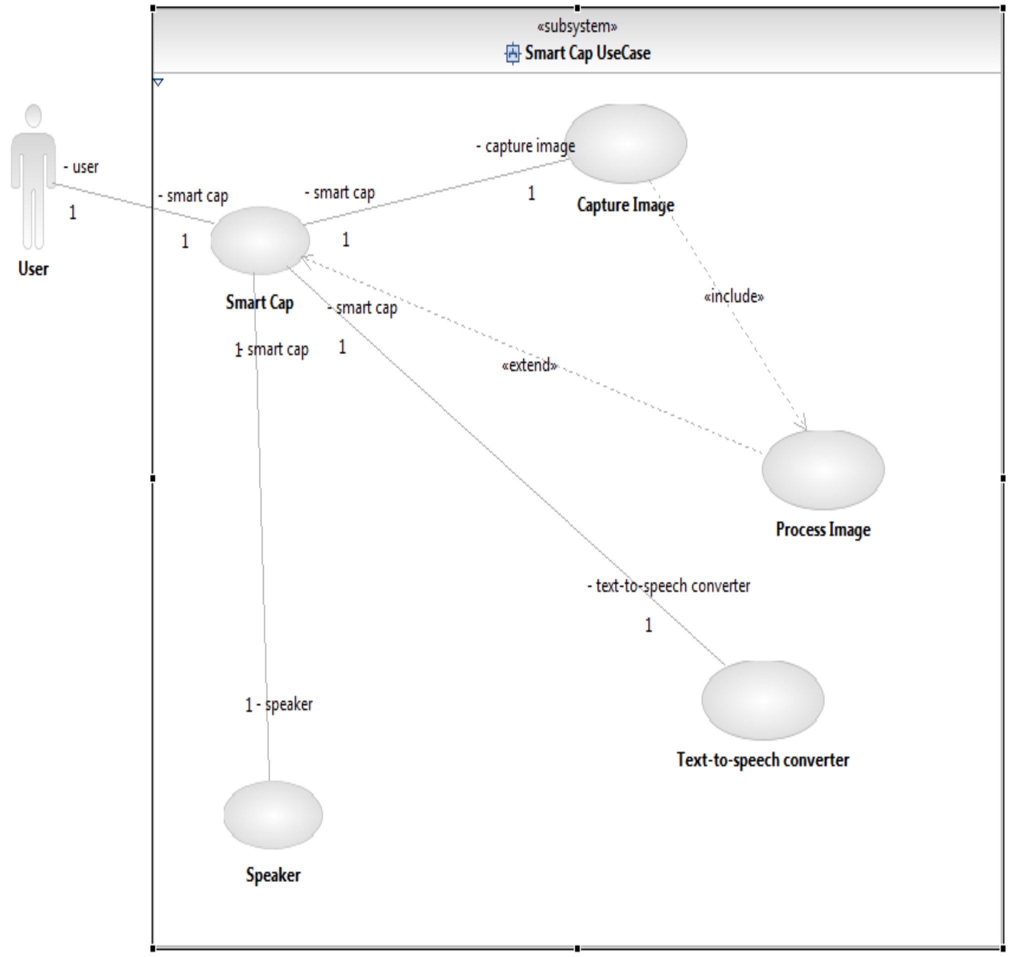


FIG 6.4.2.1 USE CASE DIAGRAM

The above diagram is Use Case diagram of our system. It shows the set of actions performed by various users. In our system we have our user As described earlier, the content in the Ovals areactions performed in the system and those actors are like symbols represent users in system. Those dashed lines from user to action means users are performing those actions respectively.

2. SEQUENCE DIAGRAM

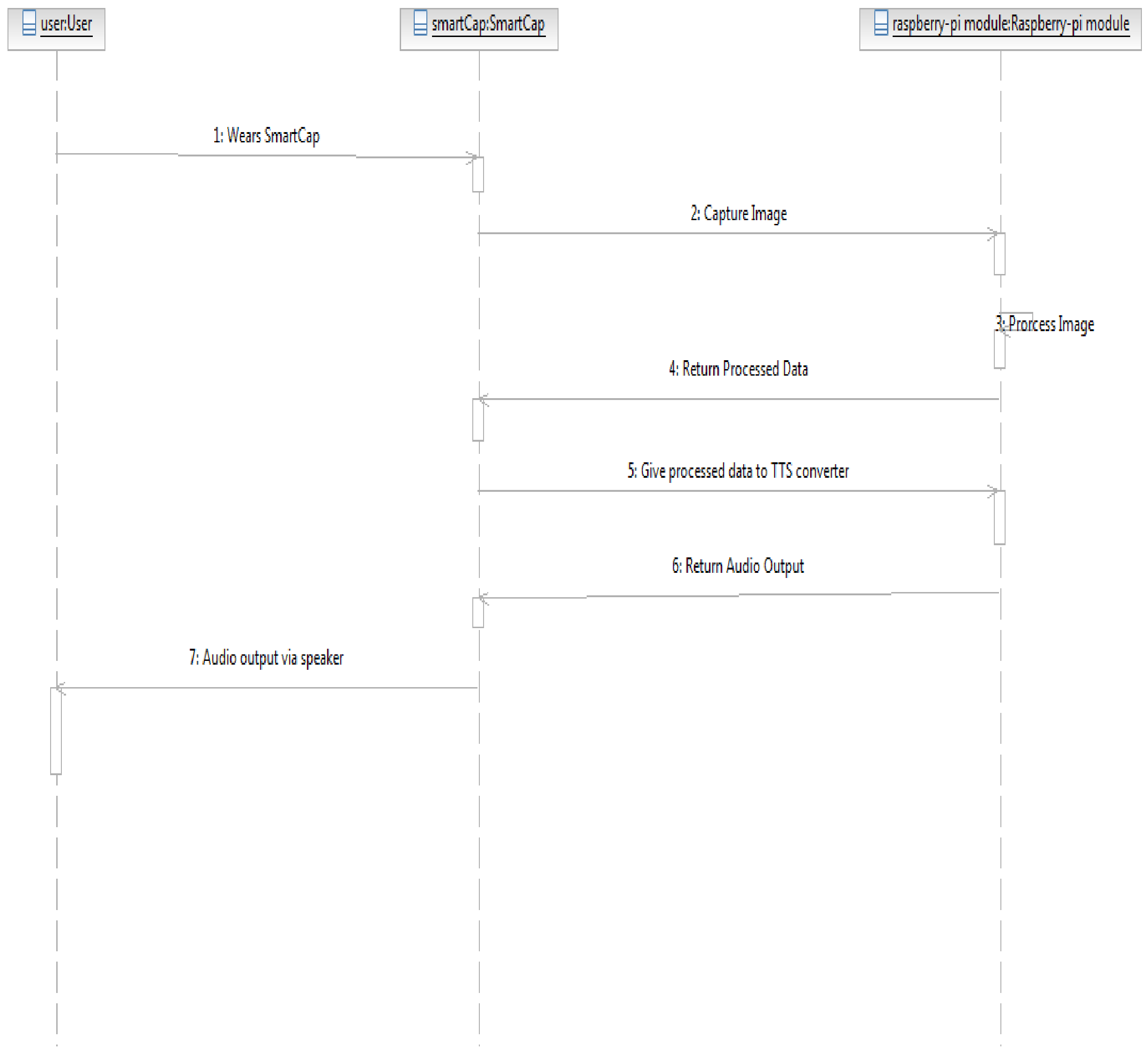


FIG 6.4.2.2 SEQUENCE DIAGRAM

The above diagram Show sequence diagram for an user. It represents sequence or flow of messages in system among various objects of the system in user's life time. The rectangle boxes at top represent objects that are invoked by unregistered user and the dashed lines dropping from those boxes are life lines which shows existence of the object up to what time. The boxes on the dashed lines are events and the lines connecting them represent messages and their flow.

3. CLASS DIAGRAM

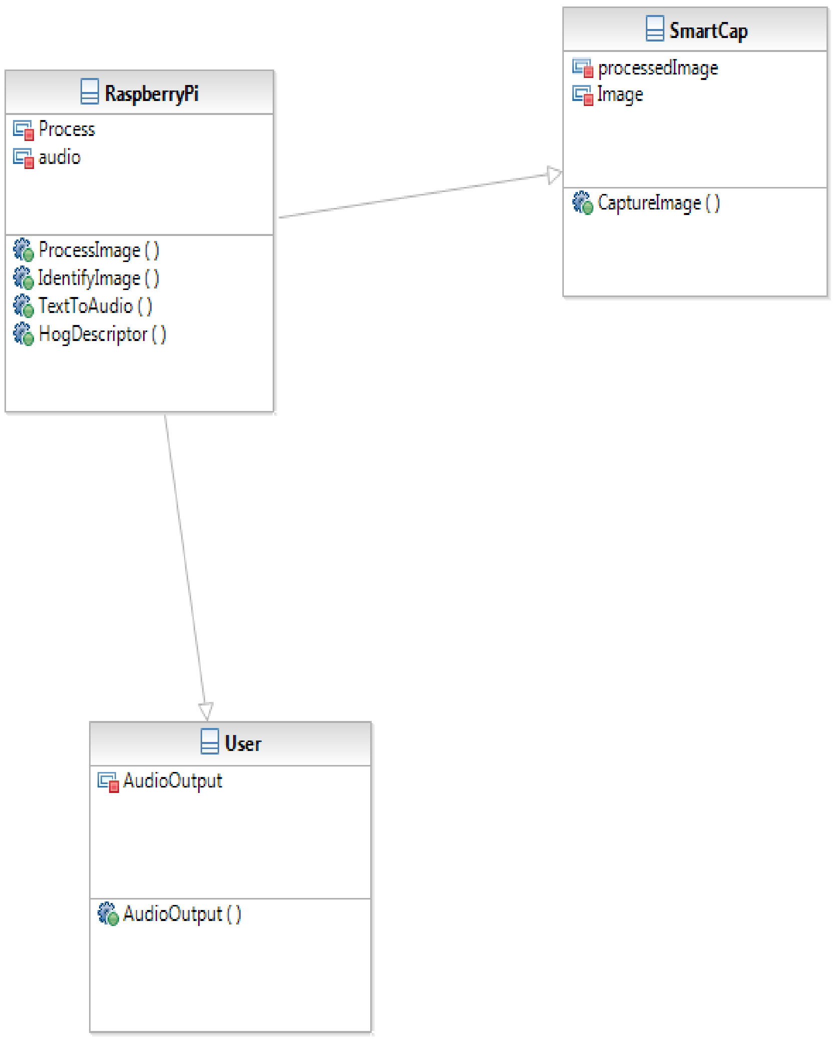


FIG 6.4.2.3 CLASS DIAGRAM

The above diagram represents class diagram of our system i.e., it shows various classes used in our system and the relationship with one class to other in the system. Each rectangle box represents a class and the upper portion of it represents class name and middle portion represents attributes of the class and the lower represents the functions performed by that class.

4.ACTIVITY DIAGRAM

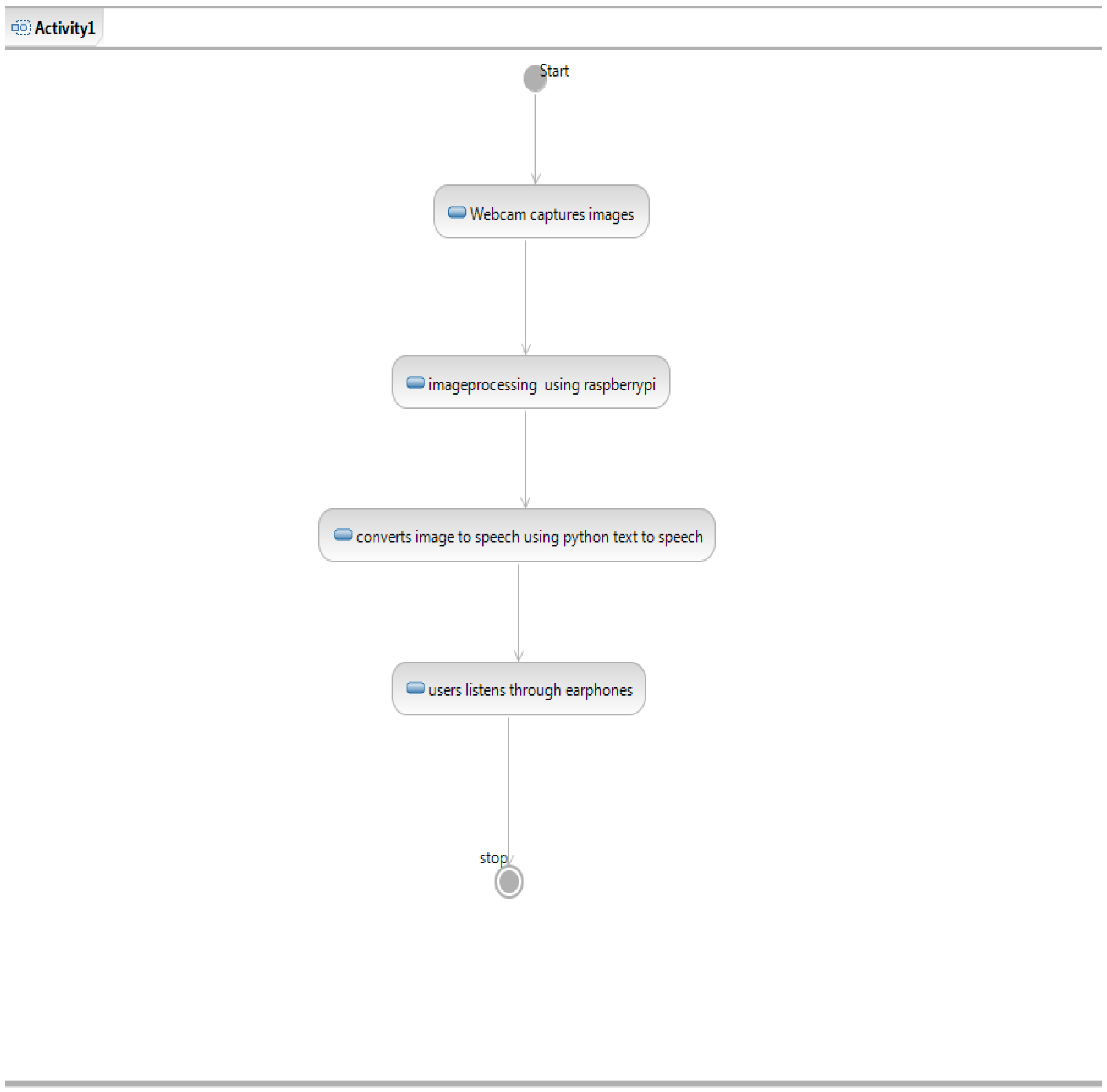

Activity1
Start
Webcam captures images
imageprocessing using raspberrypi
converts image to speech using python text to speech
users listens through earphones
stop

FIG 6.4.2.4 ACTIVITY DIAGRAM

The above diagram represents activity diagram of the system i.e., it represents the flow of activities in our project Organized User Search Histories. Dot at the start represents starting and dot with circle represents ending and an activity is represented as curve sided rectangle. On seeing it we can understand the flow activities that has to be gone from start to end.

5.INTERACTION DIAGRAM

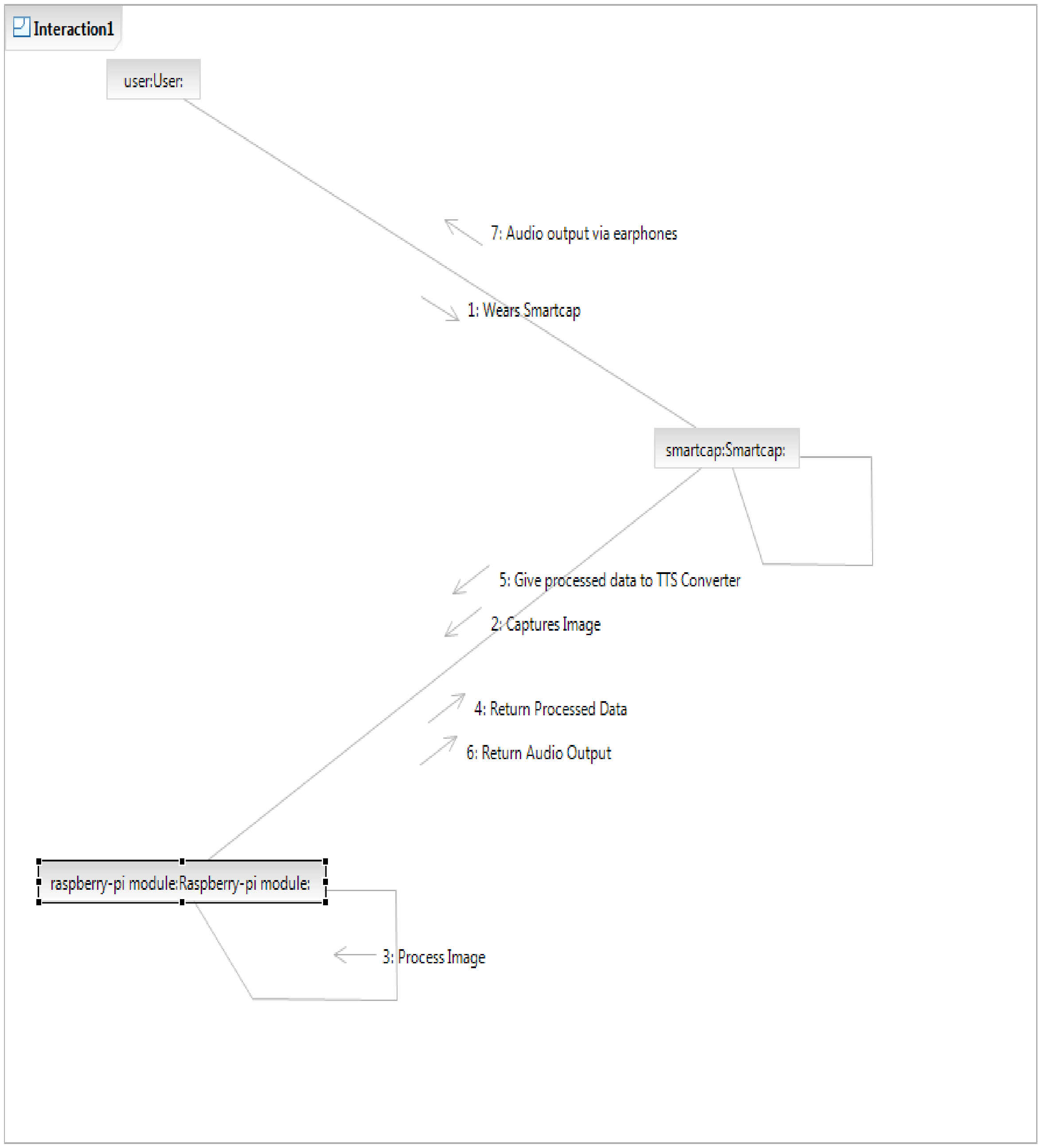
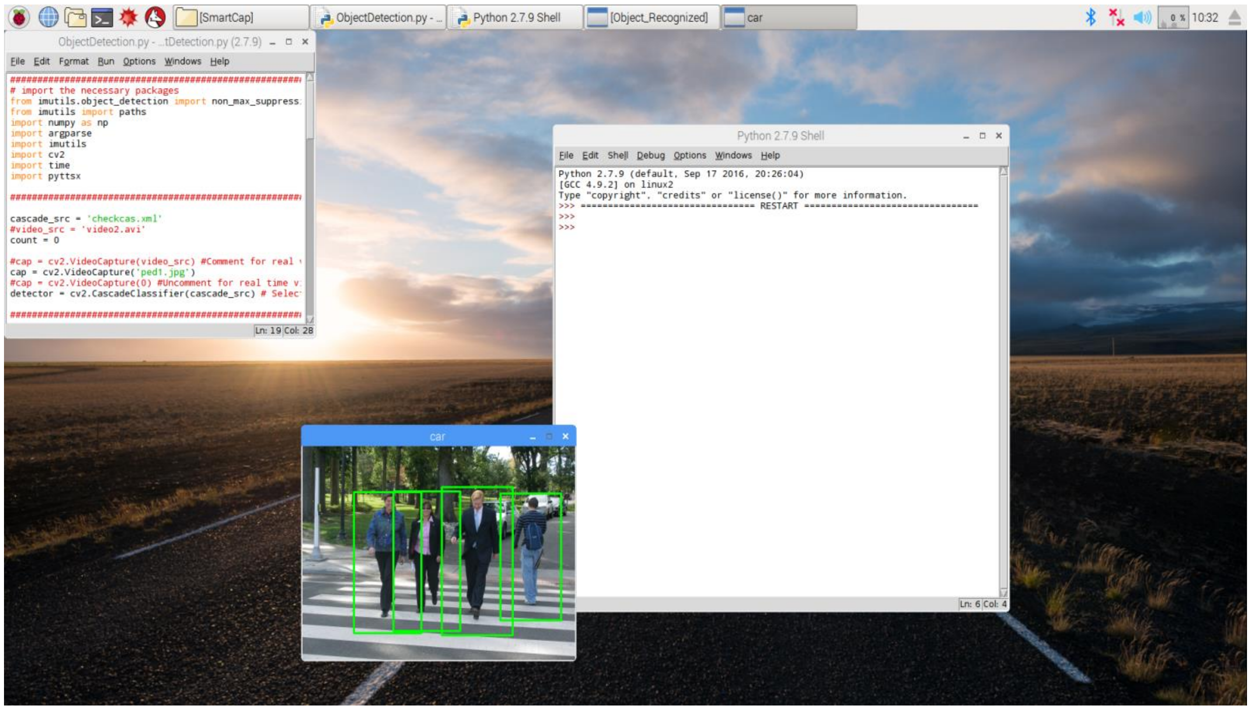


FIG 6.4.2.5INTERACTION DIAGRAM

Interaction diagram shows the series of interaction between the objects and it shows the operations done of them.

7. OUTPUT SCREENS

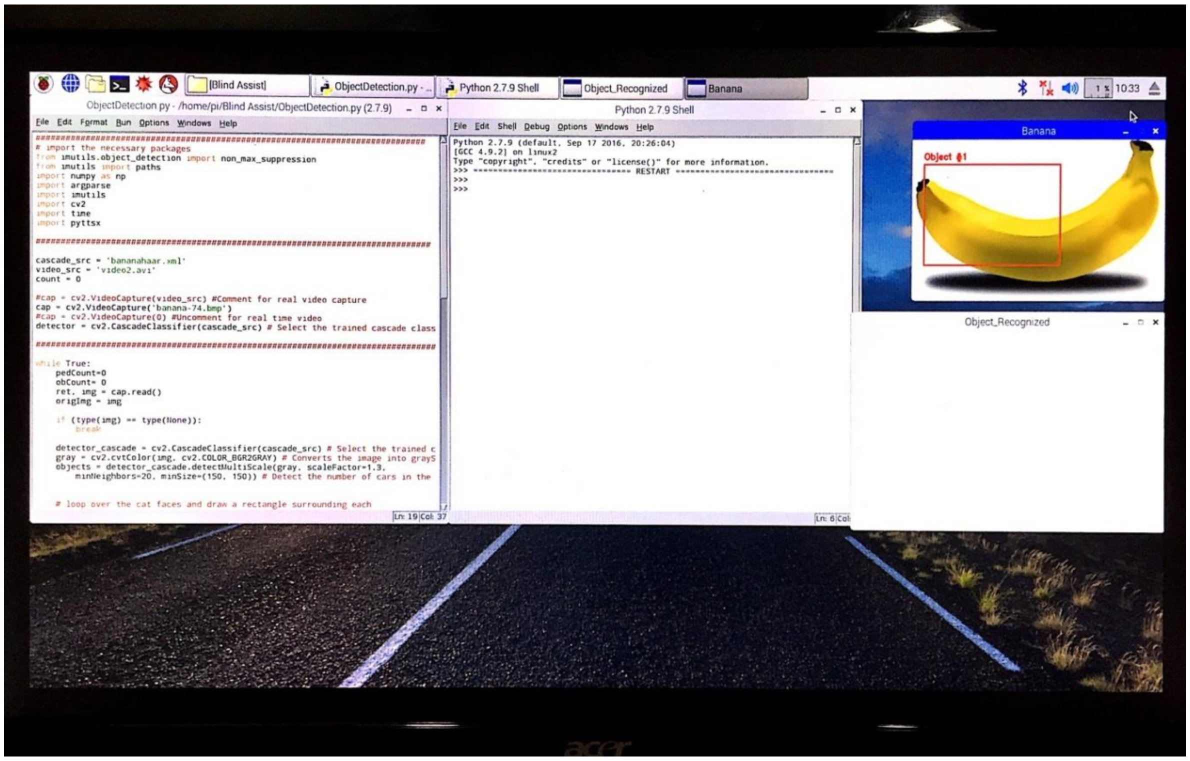

[SmartCap]
ObjectDetection.py - ...
Python 2.7.9 Shell
[Object_Recognized]
car
10:32
ObjectDetection.py - ...tDetection.py (2.7.9)
File Edit Format Run Options Windows Help
Ln: 19 Col: 28
Python 2.7.9 Shell
File Edit Shell Debug Options Windows Help
Ln: 6 Col: 4

```
##########################################################
# import the necessary packages
from imutils.object_detection import non_max_suppress
from imutils import paths
import numpy as np
import argparse
import imutils
import cv2
import time
import pyttsx

##########################################################

cascade_src = 'checkcas.xml'
#video_src = 'video2.avi'
count = 0

#cap = cv2.VideoCapture(video_src) #Comment for real
cap = cv2.VideoCapture('ped1.jpg')
#cap = cv2.VideoCapture(0) #Uncomment for real time v
detector = cv2.CascadeClassifier(cascade_src) # Selec

##########################################################
```

```
Python 2.7.9 (default, Sep 17 2016, 20:26:04)
[GCC 4.9.2] on linux2
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
>>>
```

FIG 7.1:Pesdestrian deduction program output

- In the above output the pedestrians are detected and the output is generated as green rectangular boxes on the screen and even the audio is also produced thus helping in assisting the blind

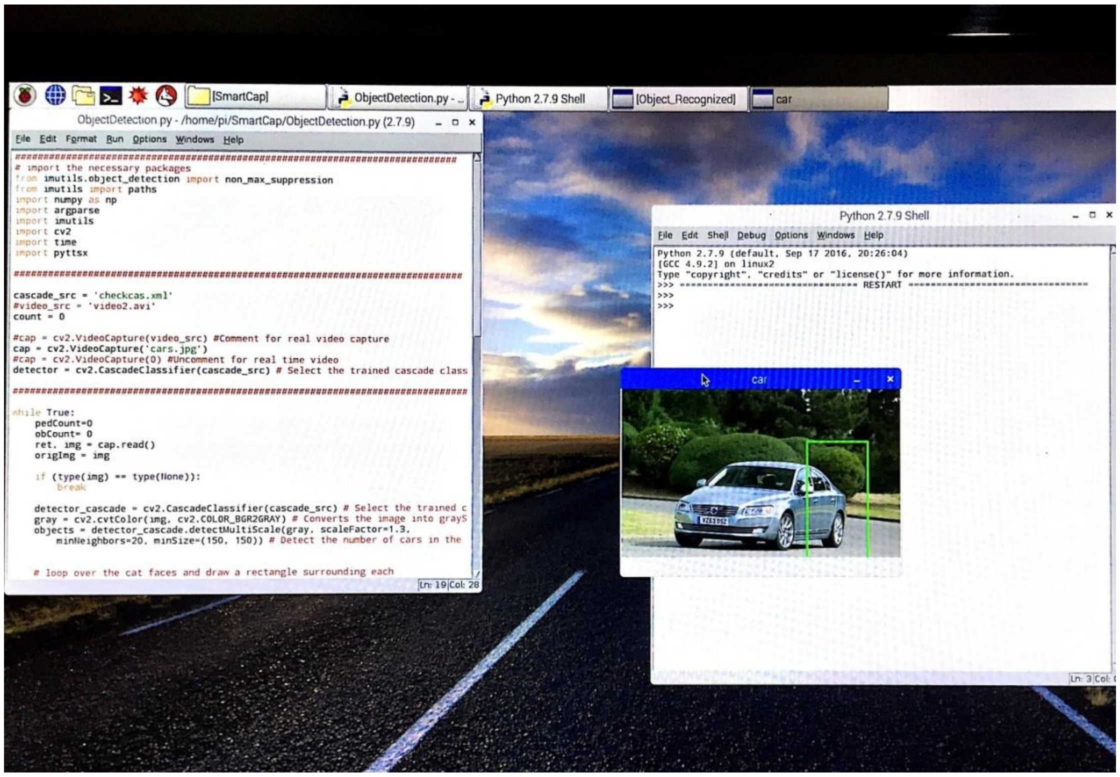

[Blind Assist]
ObjectDetection.py - ...
Python 2.7.9 Shell
Object_Recognized
Banana
1033
ObjectDetection.py - /home/pi/Blind Assist/ObjectDetection.py (2.7.9)
File Edit Format Run Options Windows Help
File Edit Shell Debug Options Windows Help
Object #1
Ln: 19 Col: 37
Ln: 6 Col

```
################################################################################
# import the necessary packages
from imutils.object_detection import non_max_suppression
from imutils import paths
import numpy as np
import argparse
import imutils
import cv2
import time
import pyttsx

################################################################################

cascade_src = 'bananahaar.xml'
video_src = 'video2.avi'
count = 0

#cap = cv2.VideoCapture(video_src) #Comment for real video capture
cap = cv2.VideoCapture('banana-74.bmp')
#cap = cv2.VideoCapture(0) #Uncomment for real time video
detector = cv2.CascadeClassifier(cascade_src) # Select the trained cascade class

################################################################################

while True:
    pedCount=0
    obCount= 0
    ret, img = cap.read()
    origImg = img

    if (type(img) == type(None)):
        break

    detector_cascade = cv2.CascadeClassifier(cascade_src) # Select the trained c
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) # Converts the image into grayS
    objects = detector_cascade.detectMultiScale(gray, scaleFactor=1.3,
        minNeighbors=20, minSize=(150, 150)) # Detect the number of cars in the


    # loop over the cat faces and draw a rectangle surrounding each
```

```
Python 2.7.9 (default, Sep 17 2016, 20:26:04)
[GCC 4.9.2] on linux2
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
>>>
```

FIG 7.2:Object deduction program output

In the above image the object of an banana is being detected .usually if any other object rather than the one detect in placed it doesn’t identify it.

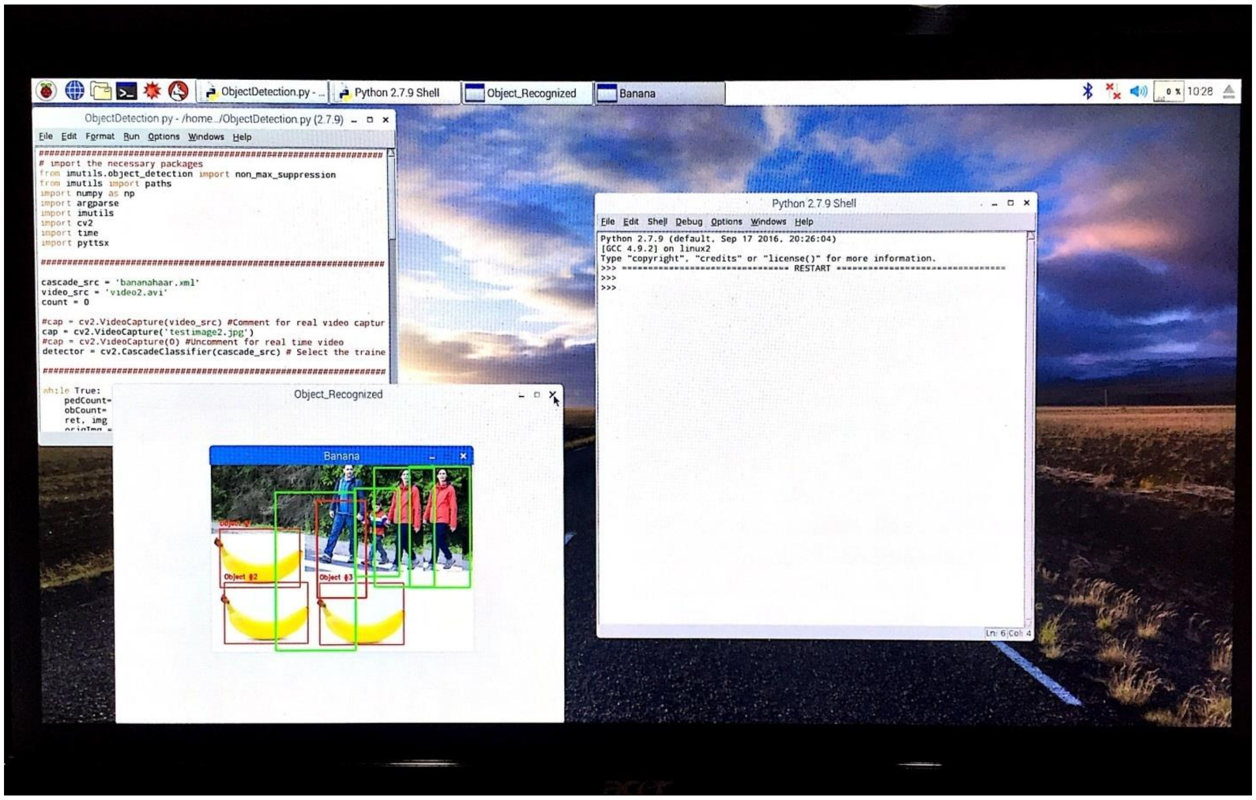

[SmartCap]
ObjectDetection.py - ...
Python 2.7.9 Shell
[Object_Recognized]
car
ObjectDetection.py - /home/pi/SmartCap/ObjectDetection.py (2.7.9)
File Edit Format Run Options Windows Help
Ln: 19 Col: 28
Python 2.7.9 Shell
File Edit Shell Debug Options Windows Help
car

```
################################################################################
# import the necessary packages
from imutils.object_detection import non_max_suppression
from imutils import paths
import numpy as np
import argparse
import imutils
import cv2
import time
import pyttsx

################################################################################

cascade_src = 'checkcas.xml'
#video_src = 'video2.avi'
count = 0

#cap = cv2.VideoCapture(video_src) #Comment for real video capture
cap = cv2.VideoCapture('cars.jpg')
#cap = cv2.VideoCapture(0) #Uncomment for real time video
detector = cv2.CascadeClassifier(cascade_src) # Select the trained cascade class

################################################################################

while True:
    pedCount=0
    obCount= 0
    ret, img = cap.read()
    origImg = img

    if (type(img) == type(None)):
        break

    detector_cascade = cv2.CascadeClassifier(cascade_src) # Select the trained c
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) # Converts the image into grayS
    objects = detector_cascade.detectMultiScale(gray, scaleFactor=1.3,
        minNeighbors=20, minSize=(150, 150)) # Detect the number of cars in the


    # loop over the cat faces and draw a rectangle surrounding each
```

```
Python 2.7.9 (default, Sep 17 2016, 20:26:04)
[GCC 4.9.2] on linux2
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
>>>
```

FIG 7.3:Car deduction program output

The above image shows the identification of car.



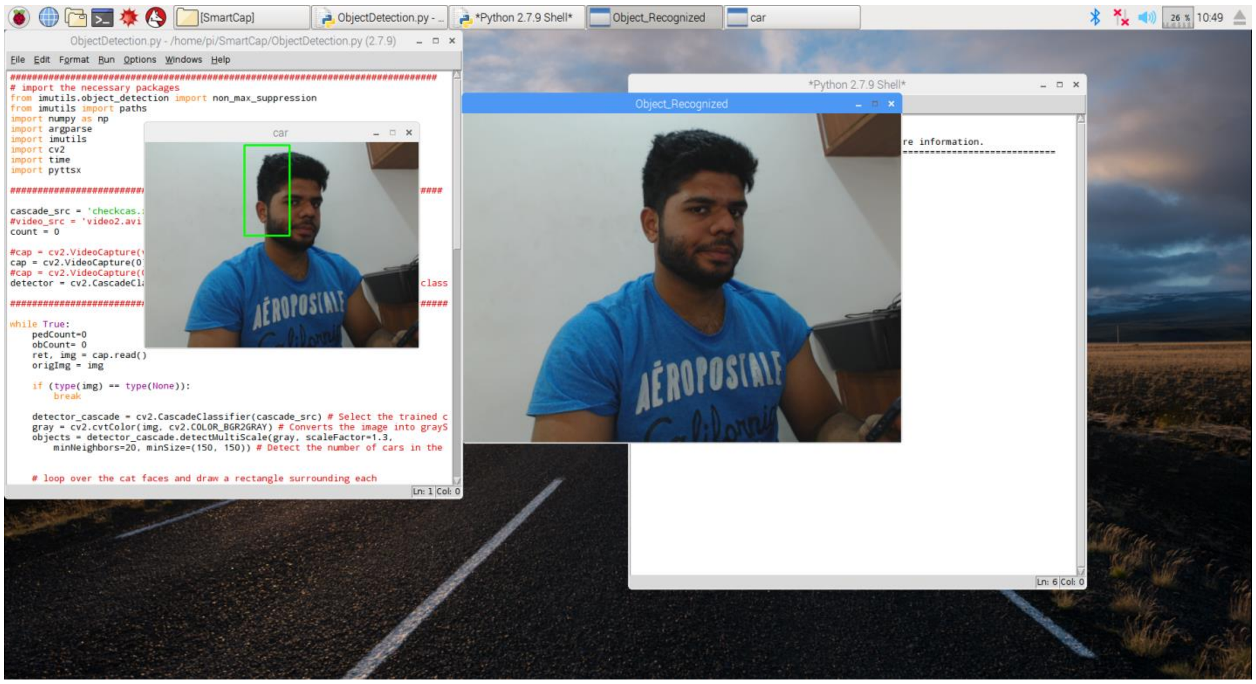
ObjectDetection.py - ...
Python 2.7.9 Shell
Object_Recognized
Banana
10:28
ObjectDetection.py - /home.../ObjectDetection.py (2.7.9)
File Edit Format Run Options Windows Help
Python 2.7.9 Shell
File Edit Shell Debug Options Windows Help
Object #2
Object #3
Ln: 6 Col: 4

```
######################################################################
# import the necessary packages
from imutils.object_detection import non_max_suppression
from imutils import paths
import numpy as np
import argparse
import imutils
import cv2
import time
import pyttsx

######################################################################

cascade_src = 'bananahaar.xml'
video_src = 'video2.avi'
count = 0

#cap = cv2.VideoCapture(video_src) #Comment for real video captur
cap = cv2.VideoCapture('testimage2.jpg')
#cap = cv2.VideoCapture(0) #Uncomment for real time video
detector = cv2.CascadeClassifier(cascade_src) # Select the traine

######################################################################

while True:
    pedCount=
    obCount=
    ret, img
```

```
Python 2.7.9 (default, Sep 17 2016, 20:26:04)
[GCC 4.9.2] on linux2
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
>>>
```

FIG 7.4:Pedestrains and object deduction program output

The output shows the image of the pedestrians and banana are detected at once


[SmartCap]
ObjectDetection.py - ...
Python 2.7.9 Shell
Object_Recognized
car
10:49
ObjectDetection.py - /home/pi/SmartCap/ObjectDetection.py (2.7.9)
File Edit Format Run Options Windows Help
Ln: 1 Col: 0
Ln: 6 Col: 0

```
###############################################################################
# import the necessary packages
from imutils.object_detection import non_max_suppression
from imutils import paths
import numpy as np
import argparse
import imutils
import cv2
import time
import pyttsx

##########################                                                 ####

cascade_src = 'checkcas.
#video_src = 'video2.avi
count = 0

#cap = cv2.VideoCapture(
cap = cv2.VideoCapture(0
#cap = cv2.VideoCapture(
detector = cv2.CascadeCl                                                  class

##########################                                                 #####

while True:
    pedCount=0
    obCount= 0
    ret, img = cap.read()
    origImg = img

    if (type(img) == type(None)):
        break

    detector_cascade = cv2.CascadeClassifier(cascade_src) # Select the trained c
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) # Converts the image into grayS
    objects = detector_cascade.detectMultiScale(gray, scaleFactor=1.3,
        minNeighbors=20, minSize=(150, 150)) # Detect the number of cars in the

    # loop over the cat faces and draw a rectangle surrounding each
```

```
re information.
=============================
```

FIG 7.5:live Face deduction program output

The above picture indicates how the face is being identified and detected via live webcam

9. CONCLUSION

Smart cap which assists the visually impaired person by narrating the scenes which are captured by webcam has many advantages for the visually impaired. It has made navigation easy for the blind people and also made navigation safe for the blind. On a whole, the smart cap makes the life of visually impaired persons lively and beautiful. Now they can know what objects are there before them like we do. But never to forget, it’s still in the early stage of development. By the end of this decade, we can hope that Smart cap has lot of functions embedded into it and serves all the blind people in this world making the world a better place for the visually impaired people.

10. BIBLIOGRAPHY

Sites Referred:

http://java.sun.com

http://www.sourcefordgde.com

http://www.networkcomputing.com/

http://www.roseindia.com/