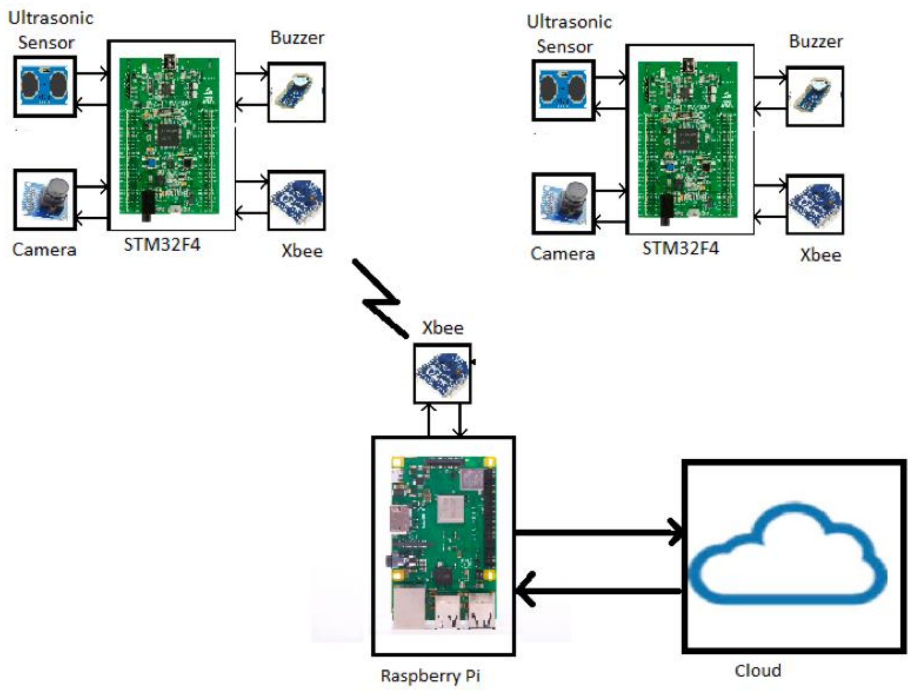
Parkware: An IOT Based Smart Parking System

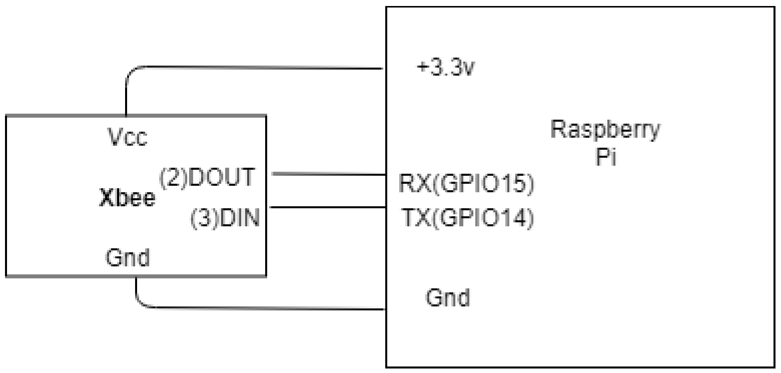
Section 1: Design Specifications

1. **Software Design Specifications**
    a. Operating System: Android OS
    b. Programming Languages for Android Application: React Native
    c. Programming Language for REST Services: Python with Flask
    d. Web Server: Apache
    e. Databases: Firebase for Real Time Data (Car Parking Map), MySql for Persistent Data
2. **Hardware Design Specifications with Quantity**
    a. Raspberry pi - 1
    b. STM32F407IGHx -2
    c. XBEE -3
    d. Ultrasonic Sensor - 2
    e. Camera -2
    f. Resistors – As per requirements
    g. LED - 2
    a. Piezoelectric buzzer – 2

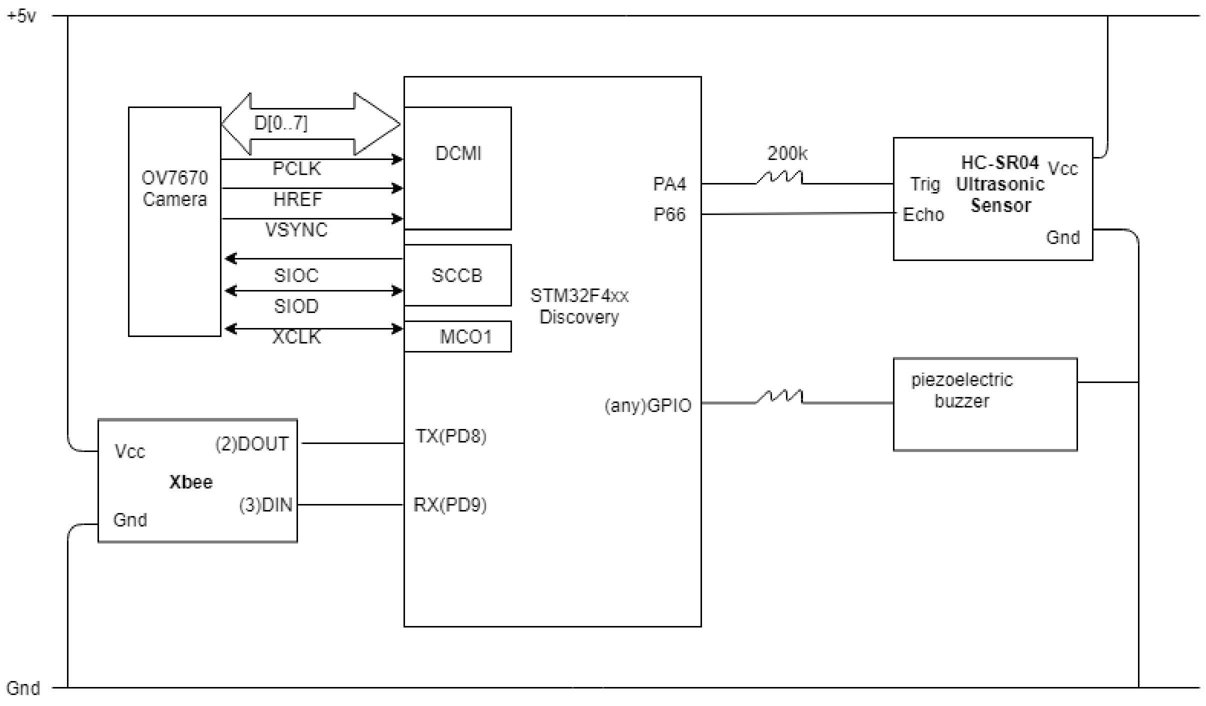


Section 2: Hardware Interfacing

a. Raspberry Pi



b. STM32F407IGHx



Section 3 : User Interface:

Android application will be made for the interface. Below are the wireframe images of the application.

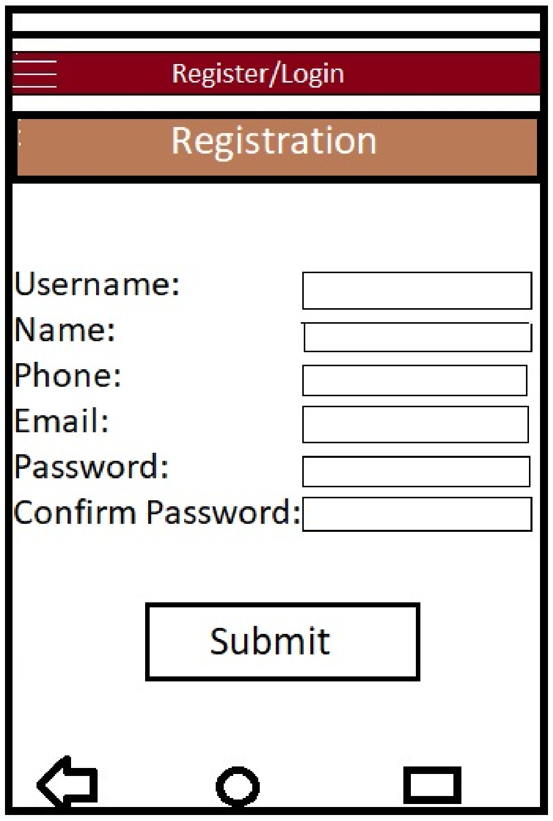


Fig. No. 3 (a): Registration

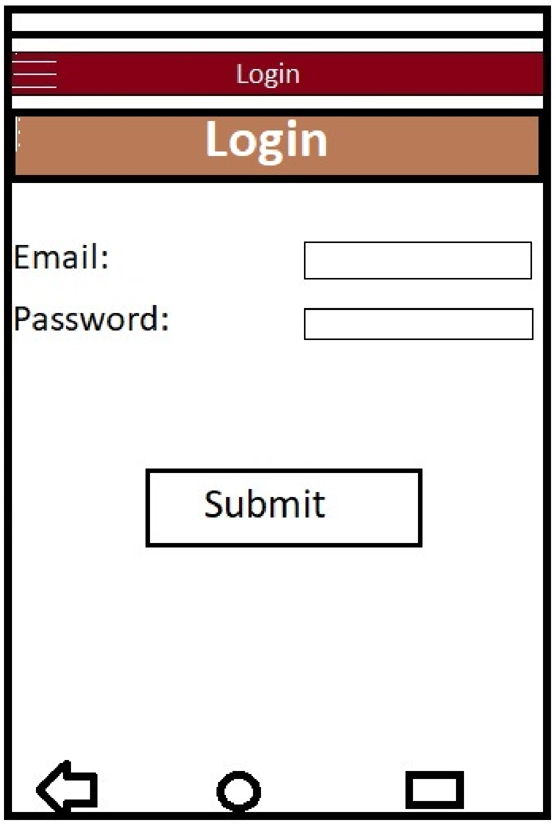
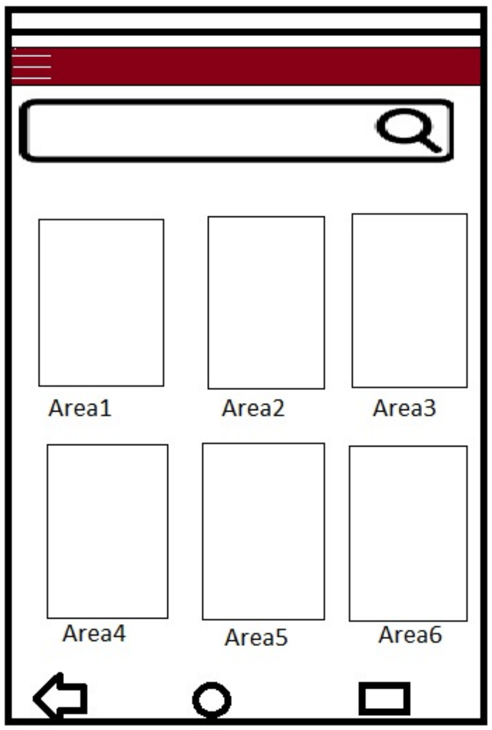
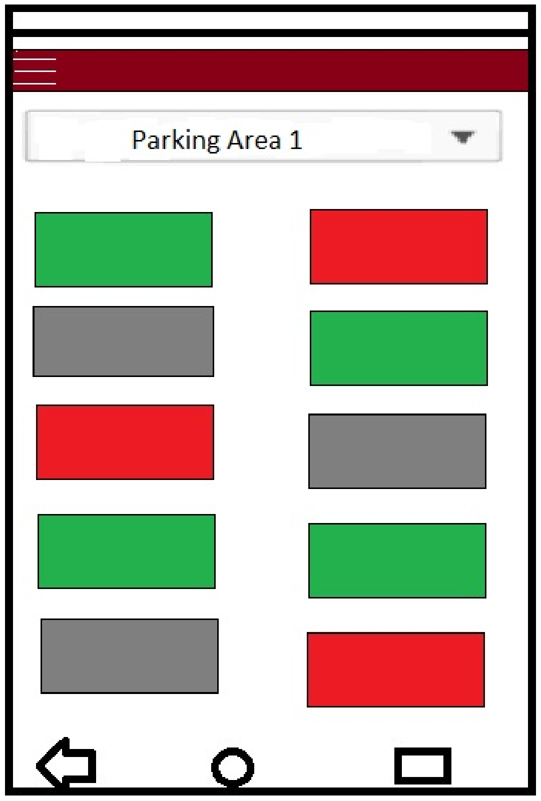


Fig. No. 3 (b): Login



This is the home screen. User will be directed to this screen upon successful Login. This screen list out all the parking area and user have an option to search the parking area by the name.

Fig. No. 3(c) Home



🟥	Unavailable
🟩	Available
▨	Reserved

Fig. No. 3(d) Parking Map

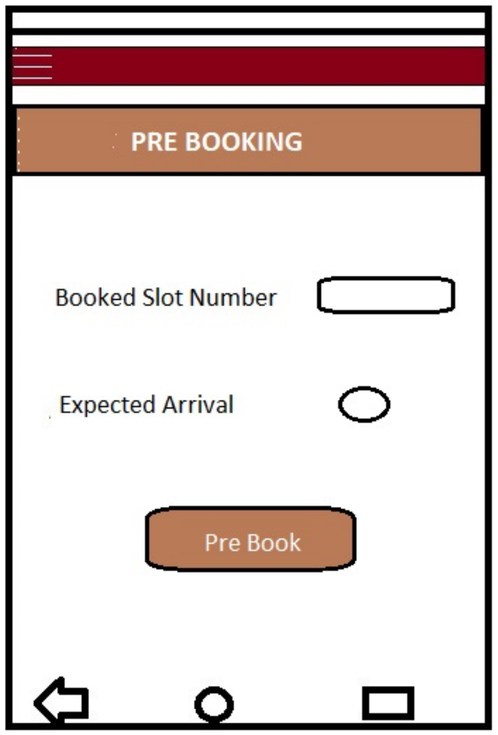


Fig. No. 3(e) Pre Booking

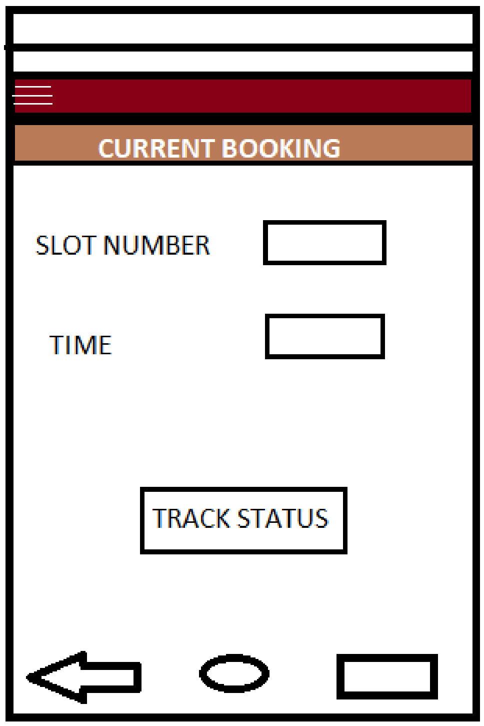


Fig. No. 3(f) Current Booking

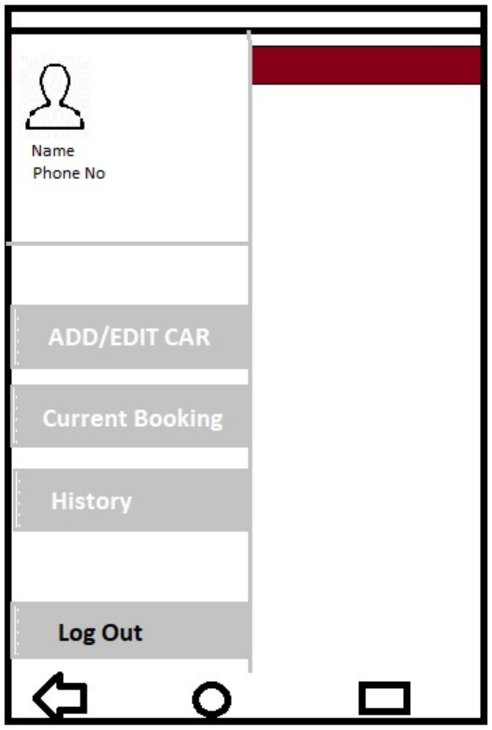


Fig. No. 3(g) Side Bar

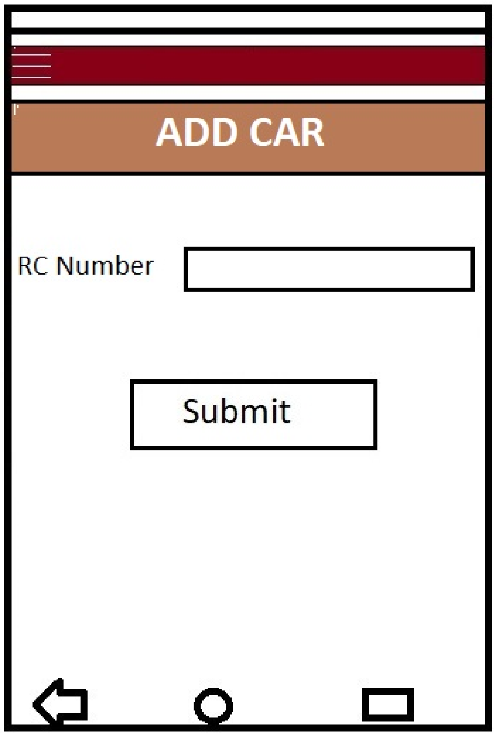


Fig. No. 3(h) Add Car

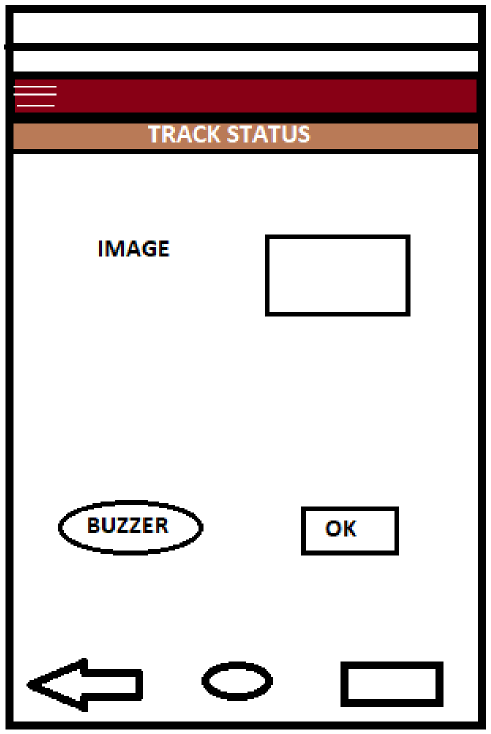


Fig. No. 3(I) Track Status

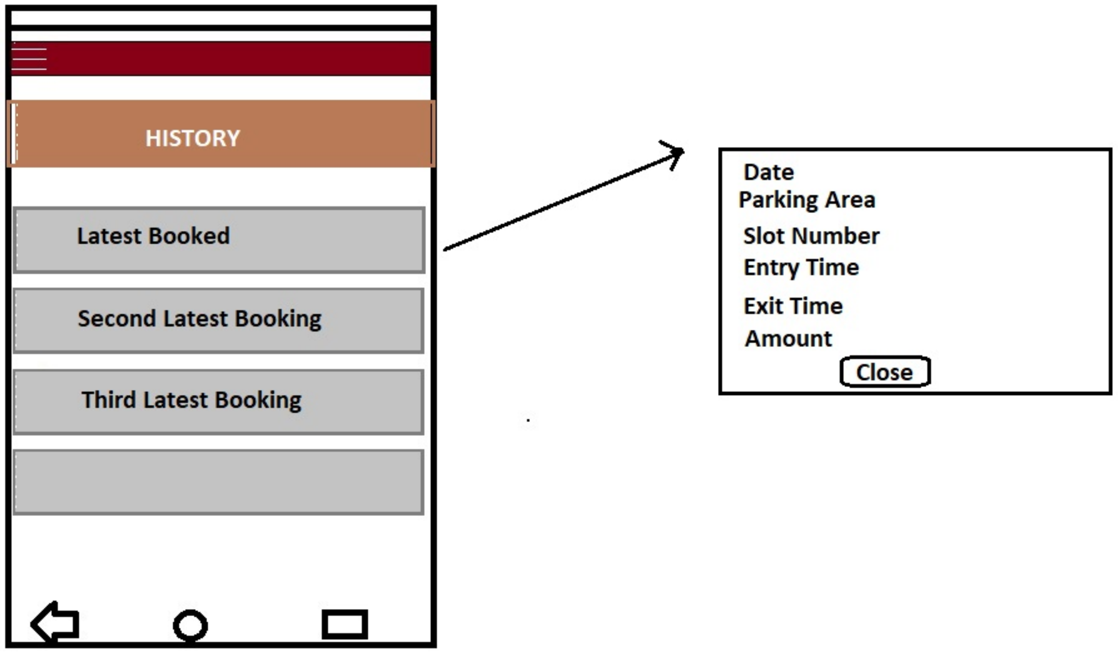


Fig. No. (j) History

Section 4 : Specification Requirement:

S1. Once the system is powered ON or if RESET is applied through push Button, STM 32 checks the connections with all 4 peripherals and connection with Raspberry Pi. If there is any issue in connection establishment then Red LED is started.

S2. User is supposed to enter all the details for the registration as per the fields in fig. 3(a). User clicks the submit button. System validates the password (combination of letters, numbers and special characters) and email and adds the user into the database.

S3. User enters email and password as shown in fig. 3(b) and presses submit button. System authenticates the credentials, upon matching of proper credentials user is logged into the system else error is shown and user is denied to login.

S4. User opens the Add/Edit Car from the side bar as shown in fig., 3(g). User enters the car registration number and submit the details as shown in fig., 3(h). System updates the user's current car number.

S5. Available list of parking areas is displayed on the user home screen as in fig., 3(c), user can select the desired parking area or filter the parking areas using search. On selection, parking map will be displayed as shown in fig., 3(d), User can check which all parking slots are available, reserved or unavailable.

S6. User selects any available slot from parking map. System redirects to pre booking screen as in fig 3(e) with already filled slot number. User selects expected arrival time and presses pre book. System updates the database with pre booking information. System automatically cancels the booking if user does not arrive in stipulated time (10 minutes after expected arrival).

S7. As soon as car enters in available parking slot, Ultrasonic sensor detects an object and sends the signal to STM32 which activates the camera, takes a snap, converts it to greyscale and forwards it to Raspberry Pi using Xbee (over zigbee network). Raspberry pi sends the image to server using REST api and on error sends the signal to STM32 to start buzzer.

S8. On receiving image from Raspberry Pi, server extract the number plate. Server checks if given parking slot is prebooked and does not match with extracted plate number then sends the error reply. If plate number matches or slot is not prebooked then, server starts the session and sends a push notification to respective user's android application.

S9. User opens the current booking screen from sidebar as shown in figure 3(f) where slot number and time of car parked will be displayed. User can press track status button which redirects user to track status screen as shown in figure 3(i). Server sends request to Raspberry Pi which forwards it to respective STM32 micro-controller. STM32 activates the camera and sends the current image back to Raspberry Pi which forwards it to server and it will be displayed to the user on track status screen.

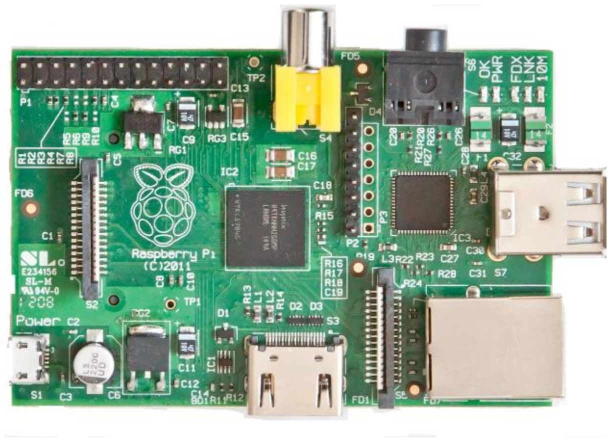
S10. User can press buzzer button on track status screen if he observes any abnormal activity. Server will request Raspberry Pi which forwards it to respective STM32. STM32 will activate the buzzer alarm for pre-defined time.

S11. Whenever a car leaves the parking spot, ultrasonic sensor detects the event and STM32 forwards it to server via Raspberry Pi. Server closes the user session, adds it to history and sends the amount due via push notification to user. Server also updates the respective parking slot as available.

S12. User can open the history screen from sidebar as shown in figure 3(j). Booking history is sorted in descending order on date.

Section 5: System Architecture

A. **Software Component**
   a. Android Application
   b. Cloud Server
   c. Web Services
B. **Hardware Components**
   a. **Raspberry pi:** A raspberry Pi is a general purpose computer which has the ability to one program at one time. There are various operating system which can run on raspberry pi, usually linux operating system is run on Rasberry pi.



   b. **STM32F407IGHx:** The STM32F4 family incorporates high-speed embedded memories and an extensive range of enhanced I/O and peripherals.

   c. **XBEE:**
      - Indoor/Urban Range: up to 100’ (30 m)
      - Outdoor line-of-sight: up to 300’ (90 m)
      - Transmit Power: 1 mW (0 dBm)
      - Receiver Sensitivity: -92 dBm
      - TX Peak Current: 45 mA (@3.3 V)

- RX Current: 50 mA (@3.3 V)
- Power-down Current: < 10 µA

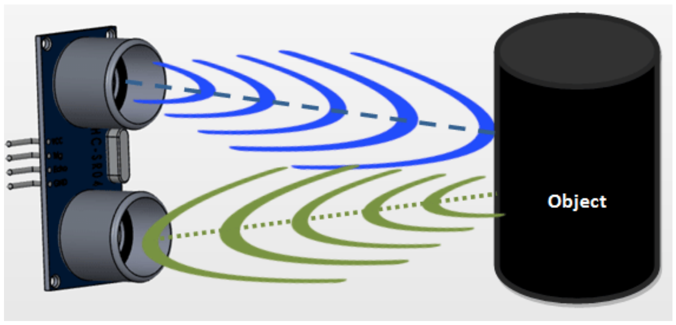
d. **OV7670 Camera:** OV7670 Camera module is small size, low voltage, single chip VGA camera and CMOS image processing module. It provides full-frame, sub-sampled or windowed 8-bit images in various formats, controlled through the Serial Camera Control Bus (SCCB) interface. The OV7670 camera module built-in onboard LDO regulator only requires single 3.3V power.

Features:

- Optic Size: 1/16 inch
- Resolution: 640x480
- Power Supply: 3.3v (inbuilt regulator)
- Lens quality: F1.8/6mm
- Sub sampling method: VarioPixel
- Pixel Density: 0.3MP

e. **HC-SR04 Ultrasonic Sensor:**

The Ultrasonic transmitter transmits an ultrasonic wave, this wave travels in air and when it gets objected by any material it gets reflected back toward the sensor this reflected wave is observed by the Ultrasonic receiver module



Reason to prefer over Infrared Sensors: ultrasonic sensor uses sound instead of light for ranging and works better in outdoor environments. Ultrasonic sensor provides a more complex signal pattern with a possibility of multiple detection

Features:

- Operating voltage: +5V
- Theoretical Measuring Distance: 2cm to 450cm
- Practical Measuring Distance: 2cm to 80cm
- Accuracy: 3mm
- Measuring angle covered: <15°
- Operating Current: <15mA
- Operating Frequency: 40Hz

f. **Piezoelectric Buzzer:** It is used whenever the operation requires a loud sound and wide frequency range.