LOCAL AIR TRAFFIC CONTROL FOR DRONES AND UAVs (LATC)

SEETA

Theme: Developer Tooling

Detecting and monitoring UAVs & Drone traffic using modular IOT gateway device solution to enhance their management and ensure security, privacy and safety of citizens using object detection, decentralized data storage and predictive analysis.

APPROACH/SOLUTION

- **Detection :** Identifying the drones & UAVs in the video feed using object detection and modular IOT gateway device solution.
- **Discovery :** Logging the identities of the drones & UAVs flying in a particular air space at any instant of time, using exchange of unique identifiers and modular IOT gateway device solution.
- **Geo-fencing :** Discovering unlawful presence and raising alarms using the detection & discovery data.
- **Monitoring :** Looking out and reporting incidents based on event detection in visual data.
- **Analysis :** Analyzing route patterns and incidents.
- **Drone Incident Reporting** : Publish drone incident reports, preventive measures and remediation using a decentralized twitter application over the Ethereum blockchain network and Embark Tools.

PROBLEM

Management of traffic on ground has been a major area of consideration that too when there is just one level of traffic. Now with increase in popularity of UAVs and drones, we will be faced with the challenges of managing a multi level traffic, monitoring safe routes to prevent aerial trespassing, detecting & preventing mid air collisions and thus limiting the exposure for the citizens.

TOOLS & TECHNOLOGY

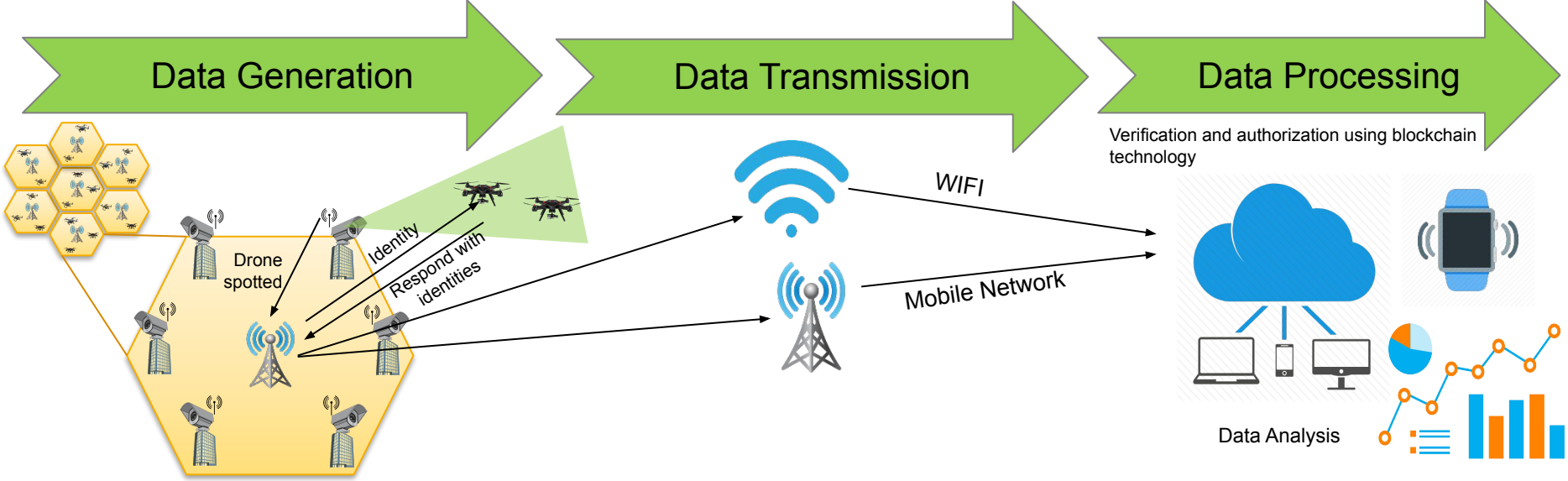
- Ethereum Blockchain, Polygon interop
- NFT.Storage
- Valist
- XMTP
- Open Source Web Spreadsheet
- IPFS distributed database
- SAP Fiori and UI5
- React.JS, Angular JS
- Coffescript, Jquery
- Node.js server, Tornado
- Nginx, Redis web server
- modular IOT gateway device

KEY IMPACT/OUTCOMES

A more safe & regulated local air traffic of drones & UAVs with availability of movement data and prediction of future of conditions. Drone incident reporting, preventive measures and remediation using a decentralized twitter application.

USING TELECOMMUNICATIONS INFRASTRUCTURE

For the purpose of discovery and logging of traffic data, we can fit every drone with a sim card and in the way as mobile towers track cell phones, in the same way the movement of the drones can be monitored. Using the concept of roaming in mobile telephony, we can detect drones that are operating in unauthorized areas.



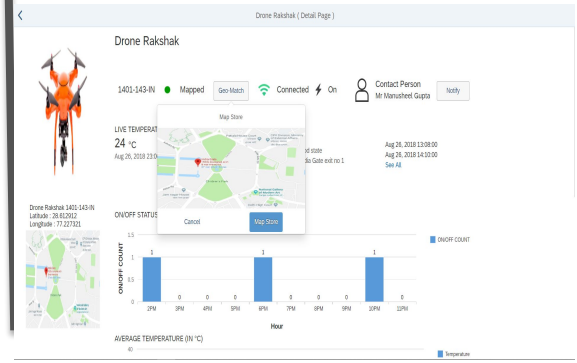
Cameras can detect the presence of a drone in the airspace using visual data. On spotting a drone, the base station present at the centre of the cell beacons all the drones in its range to send their identities and ensures that the number of detected drones is same as the number of logged drones. If a visual data shows more drones than logged drones, a warning is issued and all the logged drones are sent encrypted message to spread out so that the unlogged is singled. The warning message should be encrypted so that only the logged drones can read it.

The base station uploads the logs periodically to a central database. It sends the information about the logged drones and the warnings generated.

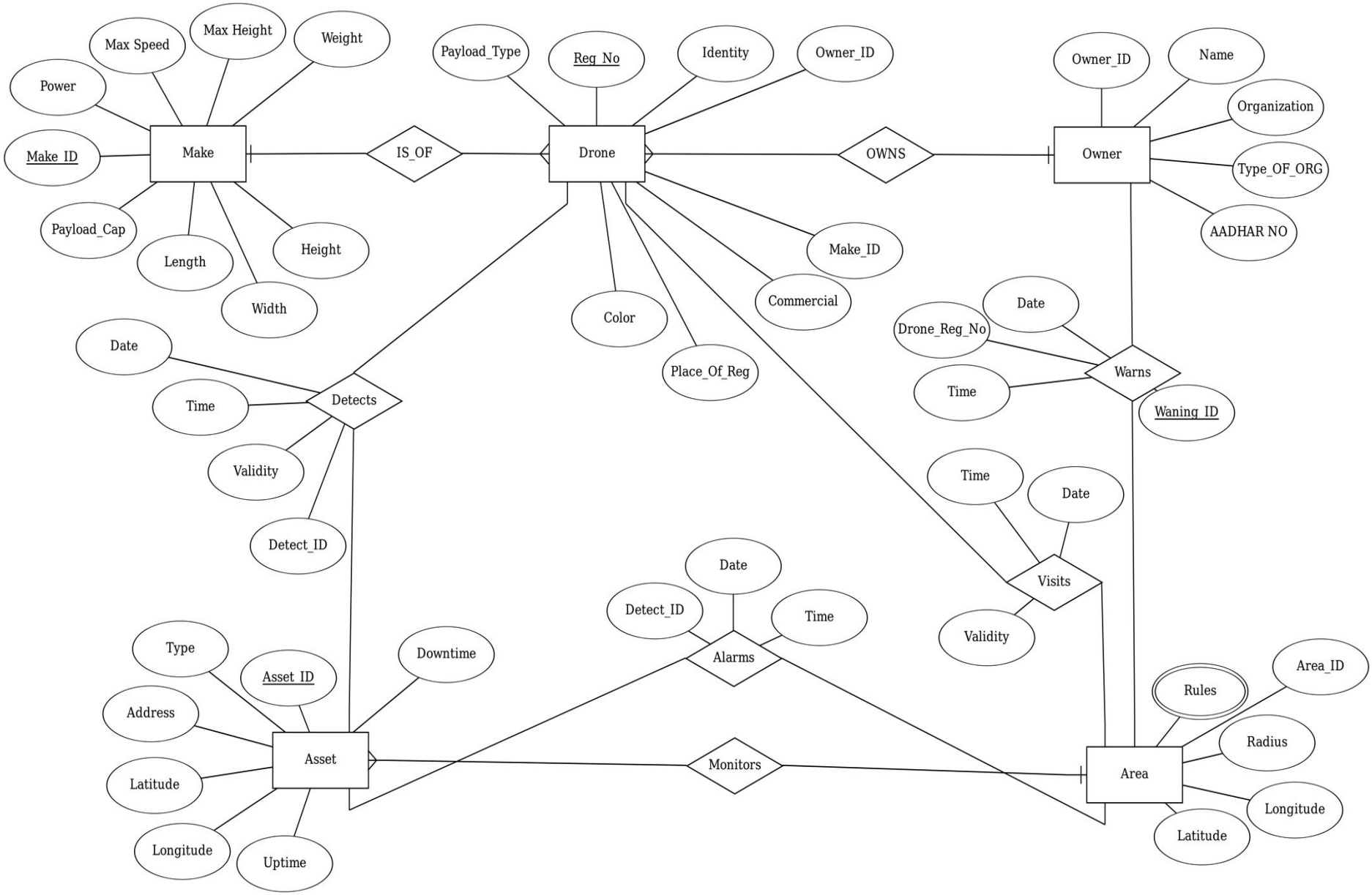
After that it cleans the logs so that it can again log the presence of allowed nodes in future. When such nodes are found whose identities do not match with the allowed identities, the warning is transmitted to its owner.

In case a silent node is found that did not log its presence, that warning is flagged as an alert and can be used to initiate further action.

Using the logs form all the cells we can obtain key insights like common routes, traffic schedule, drones demographics, alert frequency, accident hotspots. Using this information we can build predictive systems that can be used to ensure safety of drones as well as citizens.



Entity Relationship Diagram Describing the Database Setup for Drones and UAVs Traffic Management and Control



Entity Relationship Diagram Details the Database Setup for Drones and UAVs Traffic Management and Control

Each 'Area' contains a tower in the middle which communicates with drones using EM signals. An area is spherical in shape.
The diagram has the following **Entities**:
1. **Make**: This entity stores information about the construction of a drone.
2. **Drone**: This entity stores information about a drone in use including its Reg_No., Identity Code, Color, Usage Type, Payload Type etc.
3. **Owner**: This entity stores information about an owner of a drone including its name, organization, identification number
4. **Area**: A geographical area which is around a tower situated at Latitude and Longitude and having a Radius. The entity also stores the rules or patterns of a drone's Identity Code allowed in the area of a tower.
5. **Asset**: An asset is a camera situated at Latitude and Longitude. It is accesible to a tower at Address. It stores information about the uptime and downtime of a camera

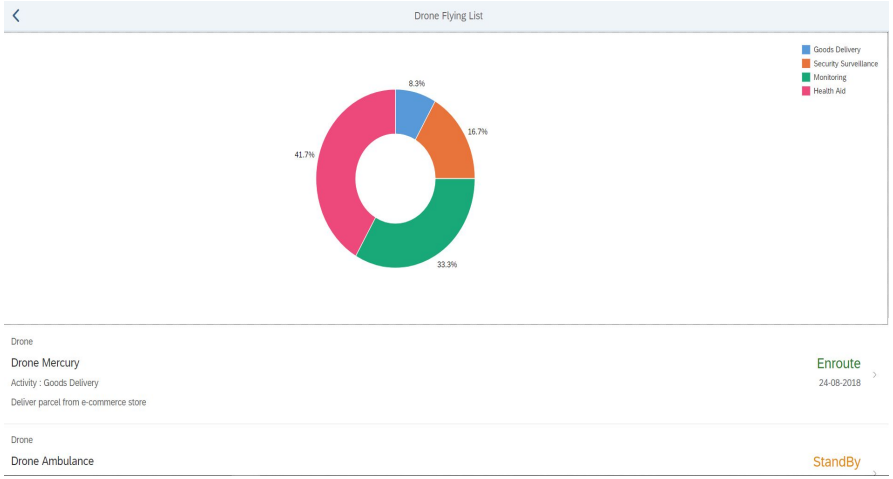
The diagram has the following **Relationships**:
1. IS_OF: Make & Drone: Specifies the make of a drone. A Drone can only have one Make.
2. OWNS: Drone & Owner: Specifies the owner of a drone. A Drone can only have one Owner.
3. Warns: Area & Owner: Stores details about warnings issued to an owner due to unlawful access by a Drone. Drone_Reg_No at Date, Time.
4. Visits: Area & Drone: It stores information about a visit of a Drone to an Area, at Date and Time and if the visit was valid or not.
5. Detects: Asset & Drone: It stores inforamaton about detection of a Drone by an Asset at Date and Time .
6. Monitors: Asset & Area: It stores inforamtion about which Asset monitors which Area. Under present scheme an Asset monitors1 Area.
7. Alarm: Asset & Area: Stores details about an Alarm raised by a tower in an Area due to an unknown drone detected by Asset at Date, Time.

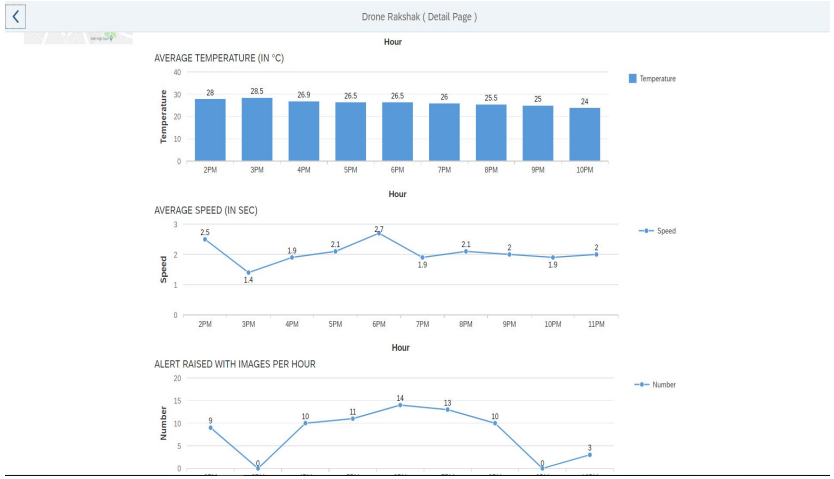
Assumptions: We assume that all the drones flying at any point of time are registered with a central authority and hence can be assigned certain unique identifiers.

How it works?

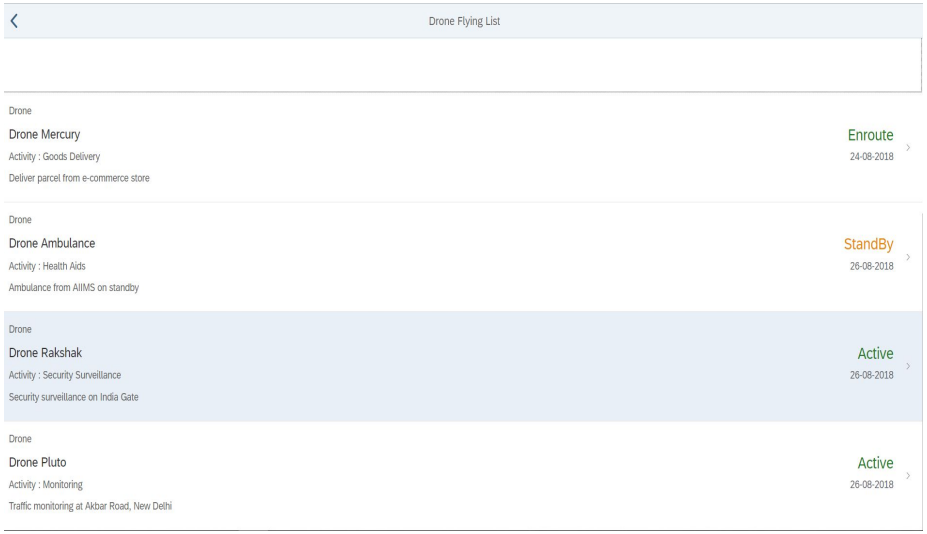
Our solution demonstrates capabilities to monitor activities of UAVs and drones using analysis of live feed from a network of cameras which are aided by detection software trained especially for drones and UAVs. To identify the drones and to provide geo-fencing our solution uses a network of base stations which periodically collect the information of UAVs and drones flying in their cells. A base station sends out a beacon to all the drones in its range to send their identity and then it matches them with its database of allowed identities. If a match to a received identity is not found then the corresponding drone and its owner are issued a warning. If a silent drone enters a cell and it is spotted by a camera but there is no log corresponding to it with the base station, then all the logged drones are issued a signal to step aside so that the silent drone can be singled out. When an operator wants to fly his/her drone in a particular areas then s/he must get his/her drone registered with the operators' base station. The above network of base stations can be conveniently emulated by the existing mobile telecommunications network and then drones fitted with sim cards can be monitored similarly as cell phones are monitored by the towers.

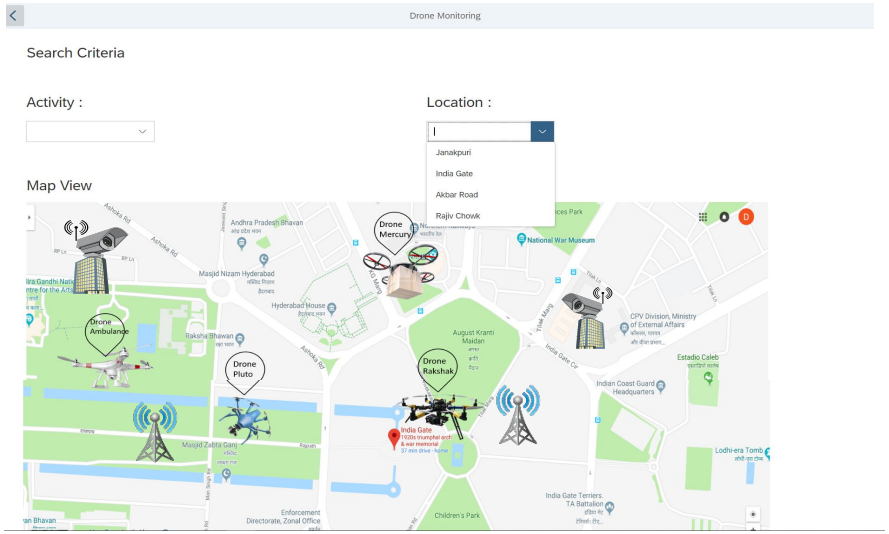
How our Magic Solution looks?





Traffic Analysis





Drone/UAV location and status

Drone Detection and Alarm Management Solution using Ethereum; National Schema for Drones and UAVs

Camera Management - add/edit/delete cameras

Click "+" icon to add cameras; Click "-" icon to delete cameras; Click "i" icon to edit cameras; Prepopulated 4 Onvif demo cameras, 10 http/rtsp cameras and 1 iOS back facing camera.

Object Detection Video Analytics Configuration

Go to Settings->Object Detection->Model to select engine; Go to Object Filters to configure selected engine object types to detect or alarm; Turn on/off detect/alarm for each object or bulk change.

Live Streaming with Object Detection Video Analytics

Implemented FFmpeg http/rtsp player; Overlays include Logo / Camera name / detected object type and location bounding boxes / Engine name and current FPS; Raise alarms in red bounding box.

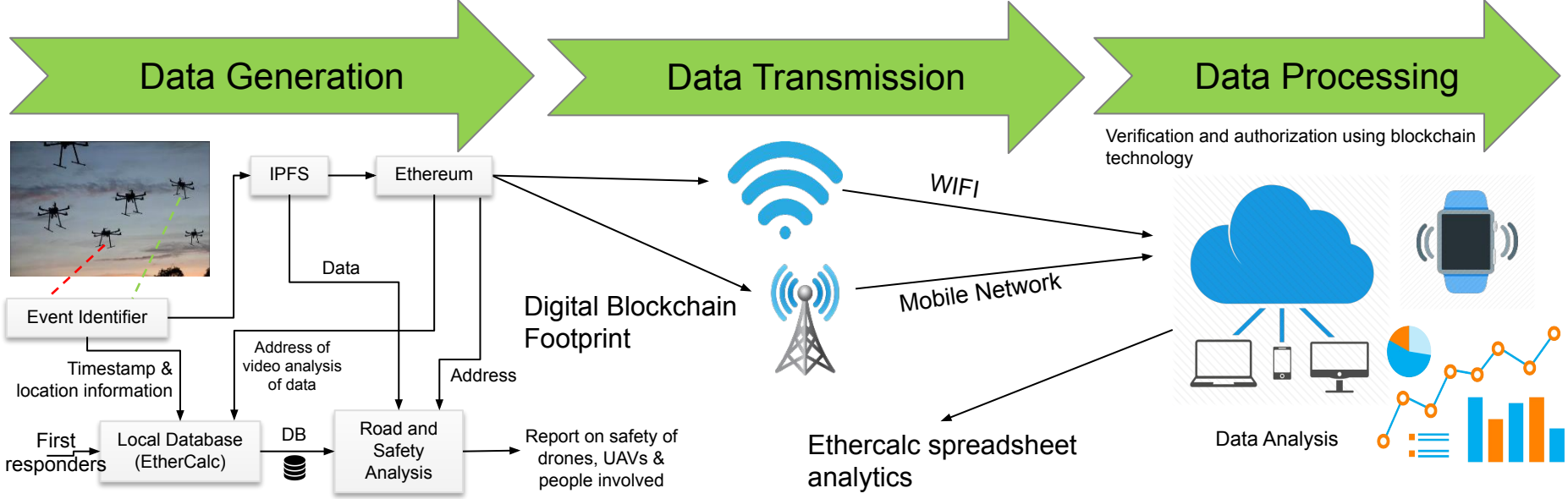
Alarm Viewer - view alarms and archive in details

Load saved alarms (green border) from IPFS/Ethereum; Metadata includes timestamp, camera name, object type and engine; Double click alarm to show in fullscreen; Select alarm (grey back color) to save.

Blockchain Alarm Storage - alarm metadata & image

Save/Delete alarm metadata and image to/from IPFS p2p distributed web; Store the hash returned from IPFS to Ethereum Test Network; Provided links to access alarms and blockchain transaction details.

National Schema for Drones and UAVs – Following are the database entries and their respective attributes:
Drone: Registration Number, Identification Code, Owner Id, Payload Type, Payload Capacity, Power, Maximum Speed, Maximum Height, Commercial, Place of Registration, Make, Color, Height, Width, Length, Weight – Details of Drone are stored in the schema.
Owner: Owner Id, Name, Organization, Organization Type, Identity Card Type, Identity Card Number – Details of Drone Owner are recorded.
Area: Area Id, Latitude of Center, Longitude of Center, Radius of Coverage – Details of the area where drone flies is recorded.
Assets: Assetid, Asset Type, Access Address, Latitude, Longitude, Area Id, Uptime, Downtime – Asset details are recorded
Drone Detected: Detected Number, Registration Number, Asset Id, Date, Time, Validity – Attributes related to the detection of drone are kept.
Drone Visited: Visit Number, Registration Number, Visit Id, Date, Time, Validity – Attributes related to the drone visit are recorded and kept.
Warnings: Warning Number, Area Id, Registration Number, Owner Id, Visit Number – Warning details for the drone are recorded and shared.
Alarms: Alarm Number, Area Id, Asset Id, Detected Number, Date, Time – Alarms for the drones are recorded, notified and logged.



Drone/UAV Incident/Traffic Management System

1. **DRIVER** : This entity contains information about the drone driver's registered information with transport authority. It includes information such as allowed drones/UAVs, nature of use, driver user ratings, driving experience etc.
This data can help us in analyzing the drivers' behavior and the sort of incidents s/he may be involved in the future.
2. **DRONES/UAVs** : This entity contains details about the drones/UAV's registry details with the transport authority. It includes the details of its make, so that the drone type involved in the incidents can be analyzed. We also store details about where and when the drone/UAV was registered so that we can analyze the drones/UAVs by states and districts. We also analyze the behavior related to the drone/UAV age.
3. **MAKE** : It stores information about the model of drones/UAVs available in the market. This helps us in analyzing incidents from the perspective of the type and design of the drone/UAV involved.
4. **LOCATION** : This entity describes locations where one or more incidents has been reported.
We include information about the characteristic of geographies around that location like nearby mountains, hills, plateaus etc
RESPONSE TEAM : This entity contains information about the response team set up, its type and volunteer count and its contact information including location. Whenever an incident is reported the nearest response team can be found using the coordinates of the location and the coordinates of the Response Team station.
The diagram contains the following relationships :
1. **INCIDENTS(Drone, Driver, Response Team, Location)** : This relationship contains information about an incident reported. The details are the Registration number that was involved, the drone involved and the location where the incident occurred, the response team that was assigned, type of incident(i.e. type of loss or injury), the date, time and the condition of air and weather at that time. If an accident involved multiple drones then multiple entries will exist with same incident id, different registration numbers and driver information.
2. **Reach(RESPONSE_TEAM, LOCATION)** : This relationship records information about which response team can reach what locations and how much time it takes (delay).
3. **IS_OF(MAKE, DRONE/UAV)** : This relationship records which DRONE/UAV is of what MAKE.

Applications of the project

The biggest reason of drones use is – they can fly for longer durations, unlike the manned aircraft.

Apart from these armed operations, drones are extensively used for:

1. **Weather detection** – Potentially, they could sample the atmosphere in difficult to reach, remote locations where weather data is scarce. The data received could then be integrated into prediction models – improving their resolution and reliability.
2. **3D mapping** – Analysts used stereoscopes to hunt for visual clues about enemy movements on photos that were stitched together to form mosaic maps.
3. **Monitoring wildlife** – “In particular it would be very useful to help find large gull chicks that hide in the dense vegetation on the island, using an infrared camera. It would be interesting to see if a small loudspeaker could be attached to use as a scarring method for playing alarm calls to frighten large gulls from the island in spring and autumn.
4. **Farming** – Robotic drones and sensors compile a large amount of the Big Data, farmers are using to increase agility and manage risk. One-third of the rice fields in Japan already is tended to by robotic helicopters. Drones can be cost-effective because they use fewer chemicals, but only with specialty crops and crops that grow on uneven terrain.
5. **Search and rescue** – Because the outer frame moves independently of the motor inside, it can also maintain altitude when colliding with objects. The shape also allows it to roll across ceilings and floors as needed. These drone can be cheaply made, providing an inexpensive and potentially invaluable tool in search and rescue operations.
6. **Law enforcement** – Recently, police in Tokyo established the first “drone squad” tasked with capturing nuisance drones flown by the public, as well as patrolling important government buildings.

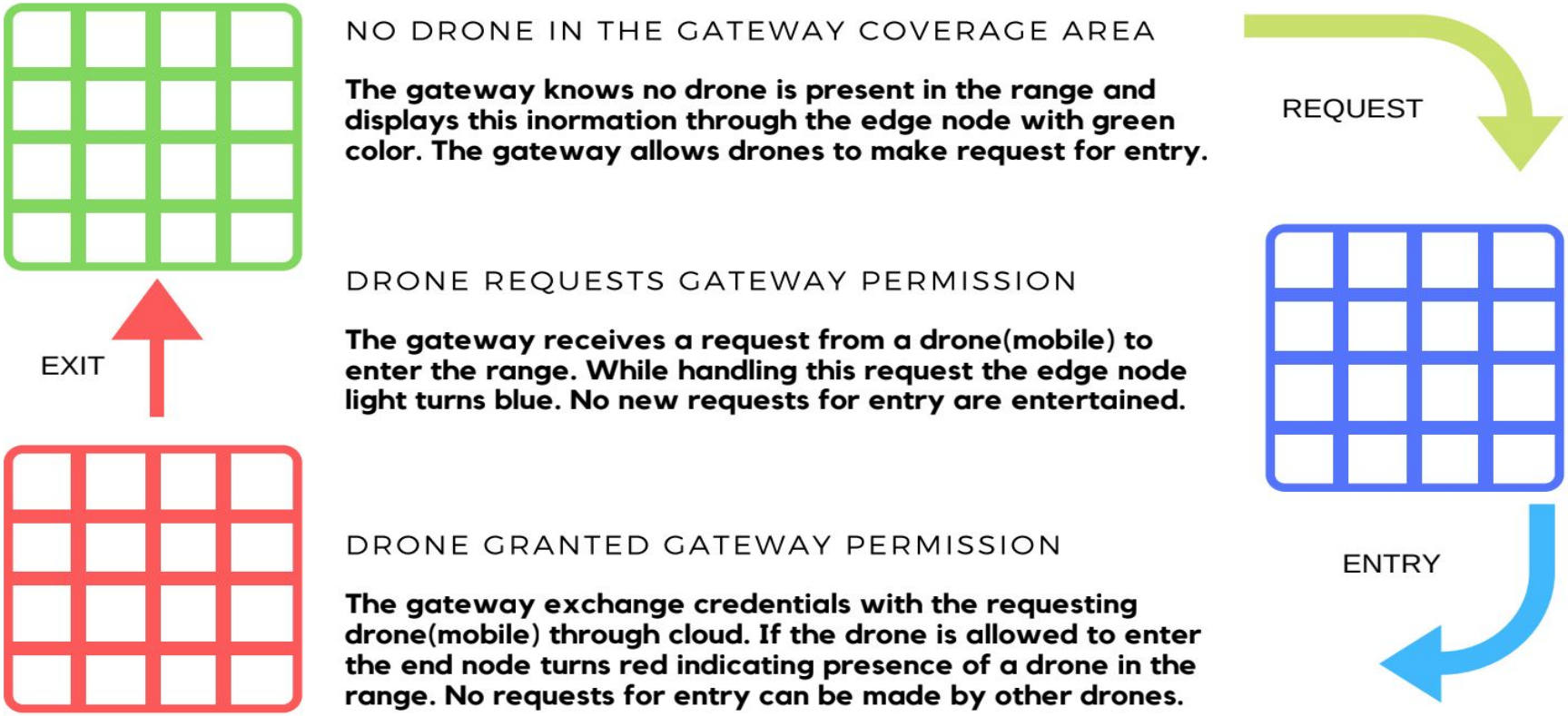
Our project aims at using the drone for the medical purposes in those areas where healthcare services are not adequate or making use of the drone to transport medical assistance which can be delayed because of heavy traffic in the region.In this we can make use of a WRLDS ball which can help the drone land at the particular location specified by the coordinates.

Limitations of the proposed project

1. **Special training:** The user needs to be given proper training before using the project.Special training is required for those going to operate the drones which add to its cost.
2. This project is semi automated.
3. **Battery life:** The battery life limits the flying time of the drone and sometimes you need to have multiple batteries fitted on the drone to extend its flying time. Cold temperatures in an area reduce the battery life.
4. **Cost:** Buying a drone fitted with all the features needed for your use may be very expensive. The federal law requires drones of different uses be fitted with specific software, hardware, and camera features and this may be costly.
5. **Weather changes:** Weather changes in an area affects the use of drones. Most drones are designed to operate at a speed of 30mph thus you cannot use them in an area with a wind speed of less than 20mph. Flying drones in rain or snow can damage the electronic components and interfere with the communication between the drone and the controller. They can only fly a few meters above the earth surface.
6. **Low operation speed:** UAVs fly at low speed making them slower compared to manned aircraft. Some drones are built to be stationary in an area like the Boeing A160T Hummingbird. Their slow speed enables them to provide accurate surveillance in an area.

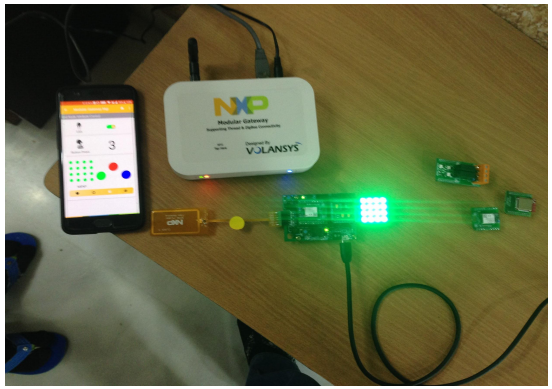
Challenges faced during implementation phase

- One of the serious issue faced by us was that even after pressing SW2 7 times, in case of Zigbee module, the next message screen did not show up as mentioned. Rather number of presses increased each time to get the **End Node Online notification.**
- Over the air update of the firmware and the mobile application could not be completed.



IMPLEMENTATION WITH NXP MODULAR IOT GATEWAY

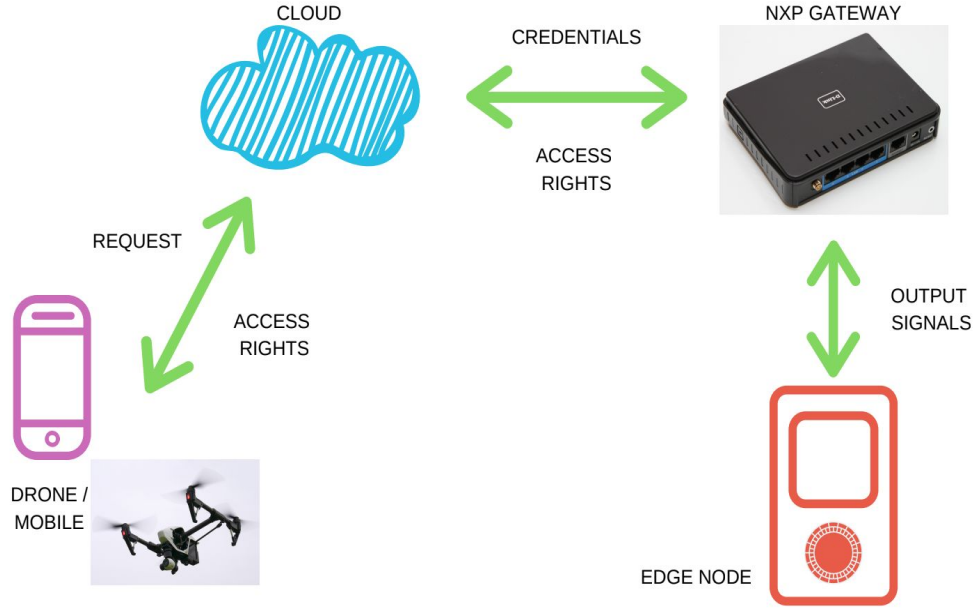
RED: The drone will stop its service and return to the base station when prompted to do so.The gateway receives a request from a drone(mobile) to enter the range. While handling this request the edge node light turns blue. No new requests for entry are entertained.



GREEN: The drone will be start its service and reach to the specified location with goods.The gateway knows no drone is present in the range and displays this information through the edge node with green color. The gateway allows drones to make request for entry.

BLUE: A sudden change of state can create chaos. So, we have decided to give some time to drone to change its state.The gateway exchange credentials with the requesting drone(mobile) through cloud. If the drone is allowed to enter the end node turns red indicating presence of a drone in the range. No requests for entry can be made by other drones.

Working Layout



COMMISSIONING

It is responsible for registration and commission of Gateway and End devices.

CLOUD MANAGER

It is responsible to connect the registered Gateway and End device to cloud.

DEVICE CONTROL

It is responsible for the Gateway and End device control/monitor. It receives control messages from the Cloud Manager and identify the control message is for the Thread or ZigBee device.

THREAD MANAGER

It is responsible to read data from thread network using UART interface connected to thread module and provide to TAP/TUN interface for application process.

ZIGBEE CONTROL

It is also responsible to data packet transmission to/from Device control to ZigBee devices.

OTA UPDATE SERVICE

OTA update service is responsible to make online software updates for gateway.

What we have finished?

1. Following steps were accomplished by us:
2. Installation of the mobile application
3. Registration and commission of the modular gateway and end devices. NFC commissioning through gateway.
4. Connection of the registered gateway and end devices to the cloud.
5. Able to access the control of the led grid both in the case of the thread and zigbee modules.
6. Simple connection of the relay click devices.

Our Team

Manu Sheel Gupta: Co-founder, Director at SEETA and Director, Aspiring Investments Corp

Manu is also Mentor, Visiting Expert at NSIT Incubation Centre funded by Delhi Government.

- Former South Asia Lead at One Laptop Per Child, Cambridge, United States of America
- Associate Product Manager at Servigistics India Office
- Co-authored over 15 research papers published in international conferences, journals
- Invited speaker at RSA Conference, San Francisco; Google, India and University of Delhi
- Education: Bachelor of Engineering in IT from NSIT, University of Delhi, India.

South Korea Collaboration: Special Award Presented to SEETA, 24th Global Contest, South Korea. The award was presented to SEETA on behalf of its remarkable results at 24th Global Software Contest hosted by IPAK and NIPA, South Korea.

Deepti Gupta: Product Engineer, Consultant and Open Source Contributor

- Developer at SAP
- Open Source Contributor

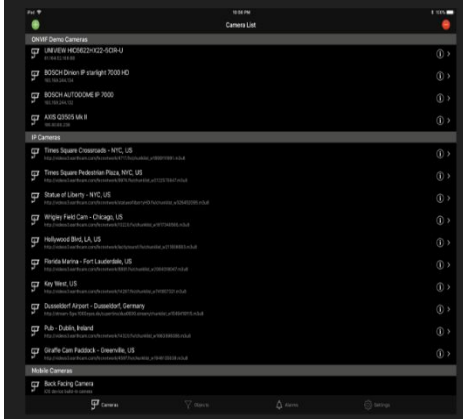
Vithika Gupta: Software Engineer, App Developer and Community Engineer

- Developer of Business and Financial products.
- Education: Computer Engineer, Banasthali University, India

SEETA

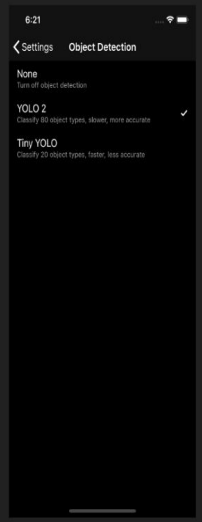
Investigative Case Management – Monitoring Cameras, Object Detection Engine for investigating cases, prevent incidents

Camera Management - camera list



- App initial view
- Three types of cameras
  - **ONVIF** - supports onvif protocol
  - **IP** - supports http/rtsp protocol
  - **Mobile** - device build-in camera
- Prepopulated 4 ONVIF cameras, 10 IP cameras and 1 iOS back facing camera
- Click "+" icon on the top left navibar to show add camera dialog
- Click "-" icon on the top right navibar to show delete icon ahead of each camera row
- Click "i" icon at the end of each camera row to show camera edit dialog
- Click camera name to start playing live streaming

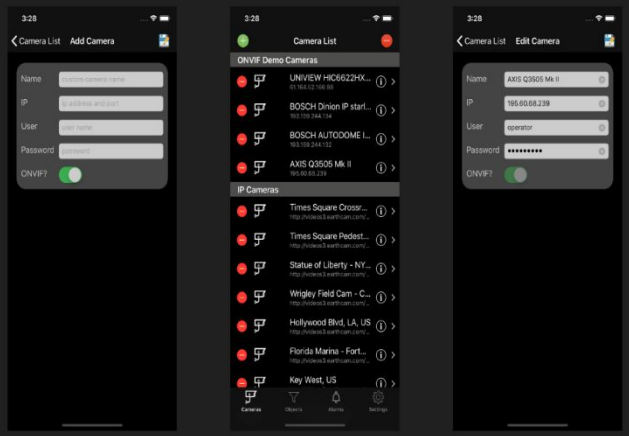
Settings - Object Detection



Three selection options

- None: show video streaming without running analytics
- YOLO 2: Object detection algorithm with pre-trained model can detect and localize 80 object types. It runs in 2-3 FPS on iPhone X or iPad Pro. Version 2 has better accuracy.
- Tiny YOLO: Object detection algorithm with pre-trained model can detect and localize 20 object types. It runs in nearly real-time (20-30 FPS) on iPhone X or iPad Pro. Tiny version has worse accuracy but faster detecting speed.
- YOLO 2 is the default option

Camera Management - add / delete / modify cameras



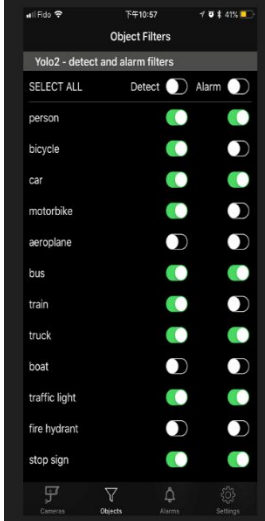
Settings

- **Object Detection**
  Select video analytics engine or turn off video analytics
- **Alarm**
  Set alarm threshold - the interval between taking alarm snapshots
- **IPFS**
  Infura remote node information
- **Ethereum**
  Wallet, balance, gas price and smart contract information
- **About**
  Author contact information

Based on the video data that is sent and suspicious objects, the Machine Learning solution can help then to do object recognition to create incidents. Details are pre-populated on analysis of the id information from the drone and added as information. An incident created is recorded in the case management solution where details/evidences etc. can be stored. Activities/e-mails and all records can be stored and retrieved from here. Linkages between the different entities can then be done to track every case.

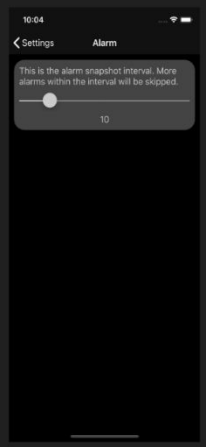
Object Detection Video Analytics

Object Detection Video Analytics Configuration



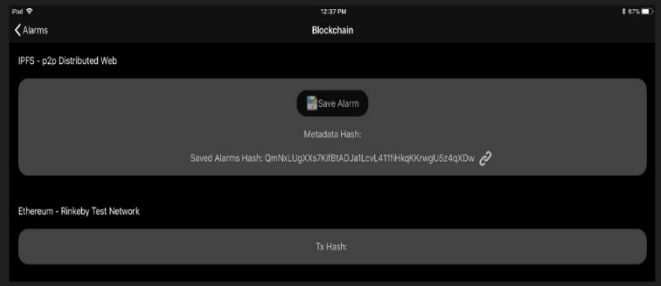
- The list content depends on the engine chosen in settings
- Yolo 2 classfies 80 object types which is listed in this diagram
- **Detect** refers to the bounding boxes and name around object on video
- **Alarm** refers to the red bounding boxes and ALARM label on object
- User can choose to turn on / off detect or alarm for each individual type
- User has option to turn on / off detect or alarm for all types
- Turn off detect would also turn off alarm automatically. If an object can't be detected, it can't be alarmed
- Detect is on by default for all object types
- Alarm is off by default for all object types

Settings - Alarm



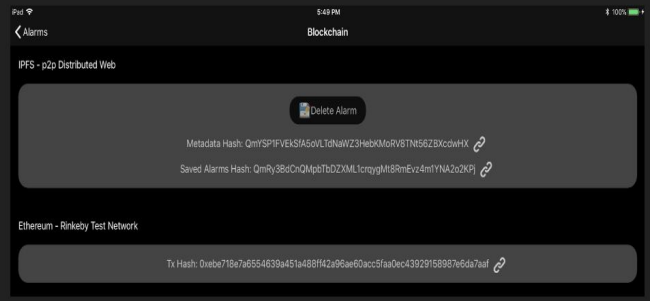
- When an alarm occurs, a snapshot is taken
- This setting is used to avoid taking too many useless snapshots. Multiple objects could be alarmed in a frame. Alarm on still object stays for every frame.
- Minimum threshold value is 1 second
- Maximum threshold value is 60 seconds
- Default threshold value is 10 seconds
- Additional alarms generated during the interval will be ignored

Blockchain - save alarm



- Click **Save Alarm** to save selected alarm image and metadata into IPFS.
- A **Metadata Hash** will be generated and displayed. Click link to view on web.
- **Saved Alarms Hash** is the link to file contains all saved alarms. It's empty when the user runs cam X first time after installation. An UUID generated as the hash key used in Smart Contract. Click link to view on web. The hash will change whenever the file content updates.
- Updated Saved Alarms Hash will be saved to Ethereum via smart contract.
- **Tx Hash** is the transaction receipt returned by Ethereum. Click link to view details.
- When the app runs next time, Saved Alarms Hash will be restored from Ethereum.

Blockchain - delete alarm



- Click **Delete Alarm** to delete selected alarm image and metadata from saved alarms file. Alarm image and metadata are permanent.
- **Saved Alarms Hash** will be changed. Click to view on web and verify selected alarm deleted from the file.
- Updated Saved Alarms Hash will be saved to Ethereum via smart contract.
- **Tx Hash** is the transaction receipt returned by Ethereum. Click link to view details.

Blockchain Transactions

Settings - IPFS Infura remote node

- Infura provides secure, reliable, and scalable access to Ethereum APIs and IPFS gateways.
- https://infura.io

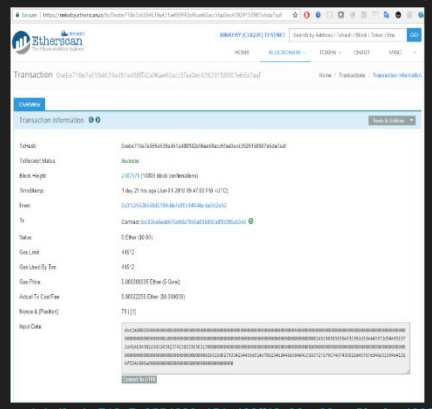
Ethereum - Smart Contract


```
pragma solidity ^0.4.24;

contract CamX {

    mapping(string => string) ipfsHash;

    function addHash(string key, string value) public {
        ipfsHash[key] = value;
    }

    function getHash(string key) public view returns (string) {
        return ipfsHash[key];
    }
}
```

DApp steps:

1. Write smart contract code in Solidity *.sol
2. Compile at https://remix.ethereum.org
3. .sol binary sent back to dapp
4. Deploy contract to network
5. Return address and abstract binary interface (ABI)
6. APIs available to be called on blockchain

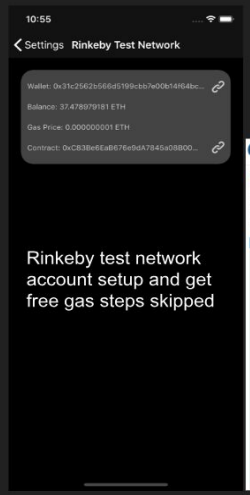
Blockchain - Ethereum smart contract transaction receipt



https://rinkeby.etherscan.io/tx/0xebe718e7a6554639a451a488ff42a96ae60acc5faa0ec43929158987e6da7aaf

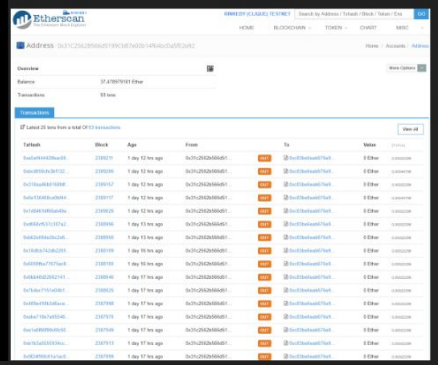
The alarm is recorded on the Blockchain Platform. This allows for an incident record to be created against every individual allowing for a“Digital blockchain footprint” to be also created for every incident reported. We uitilize IPFS and open source spreadsheet for storing details which are costly to store at Blockchain platform.

Settings - Ethereum Rinkeby test network



Rinkeby test network account setup and get free gas steps skipped

Wallet address:
https://rinkeby.etherscan.io/address/0x31c2562b566d5199cbb7e00b14f64bcda5fe2e92



Smart Contract address:
https://rinkeby.etherscan.io/address/0xc83be6eab676e9da7845a08b006a8f9d3f8a534d