I. Introduction

A Digital twin is a digital representation of a real-world physical component or an equipment. It bridges the gap between the real and virtual worlds by pairing between the digital and physical worlds using streams of data generated by sensory devices. The streams of data can vary during the lifecycle of the device and it's the combination of static and dynamic information.

II. Target Device Description (for current scenario)

A. Hardware platform
    a. Model: Raspberry Pi 4
    b. Architecture: 32 bit
B. Operating system and Programming environment
    a. OS: Raspbian
    b. Type: Debian
C. Basic Function of the Device
    a. Device imitates several full-stack devices of IT infrastructure and are extensively used in robotics, weather monitoring, task automation

III. Static information

Consists of a list of parameters which will not be going to change during the lifecycle of the device.

1. Device MAC address
2. Serial Number
3. Manufacturer: Information about manufacturer of device.
4. Hardware (Architecture information)
5. Memory Size
6. OS Info and version number:
7. Static IP (if any)
8. Price

IV. Dynamic Information:

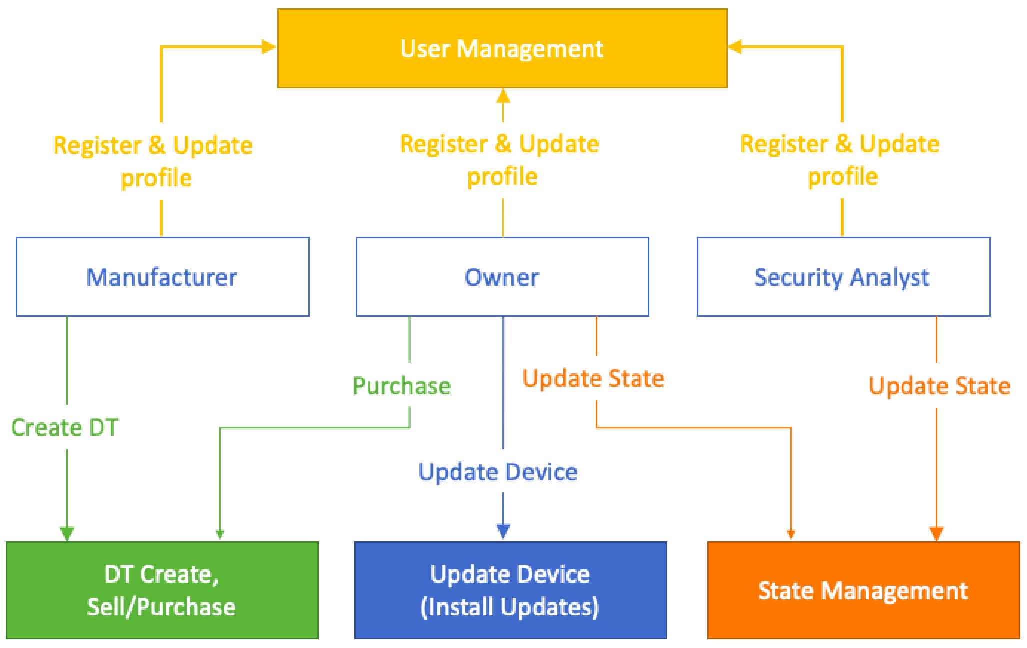
All the data/parameters which vary during the lifecycle of a device.

1. Owner Information:
    a. Owner ID
2. Location
    a. Longitude, latitude
3. Kernel Processes: **top** command can be used to show the Linux processes. It provides a dynamic real-time view of the running system. Following details about the processes running over the system can be captured.
    a. Process id
    b. Priority of the task/process
    c. Amount of shared memory used by a process
    d. Virtual memory used by the process
    e. CPU usage of process.
4. I/O latency: to monitor I/O latency in real time (if required).
5. Device uptime/last restart
6. Message buffer

    a. Message buffer of kernel
7. USB buses and the devices
    a. Bus speed
    b. Bus class
    c. Bus type
8. Peripheral Component Interconnect(PCI)
    a. PCI busses
    b. PCI devices connected
9. System Activity Report: consists of various system loads (using command: sar)
    a. Command installed using: sudo apt install sysstat
    b. CPU Usage
    c. Memory Utilisation
    d. I/O device consumption
    e. Network stats
    f. Disk Usage
    g. Process and thread allocation
    h. Battery performance
    i. Plug and play devices
    j. Processor performance
    k. Swap Memory stats
    l. Context switching stats
    m. Paging Stats
10. Interprocess Communication: Inter-process communication facilities for which the calling process has read access. Can be accessed using *ipcs* command.
    a. Message Queue
    b. Semaphore
    c. Shared memory
11. Network configuration and activity:
    a. Routing tables
    b. Interface statistics
    c. Network connections
    d. TCP ports
    e. TCP connections
    f. UDP ports
    g. UDP connection
    h. Listening ports
12. Shared object dependencies and libraries by each different processes
    a. Used direct dependencies
    b. Unused direct dependencies
13. IP layer Error reporting protocol
    a. ICMP packet type
    b. ICMP Packet code
    c. Packet source, destination
14. Address resolution (ARP) cache state
    a. Static ARP entries
    b. Dynamic ARP entries
15. Address resolution (ARP) table state
16. DNS cache state

17. SSL parameters for packets
18. Server IP
    a. Server's digital certificate
    b. Cryptographic algorithm used
19. Sensor data (device specific)
    a. Temperature, Wind speed, Moisture data (for soil sensors), Rotation speed

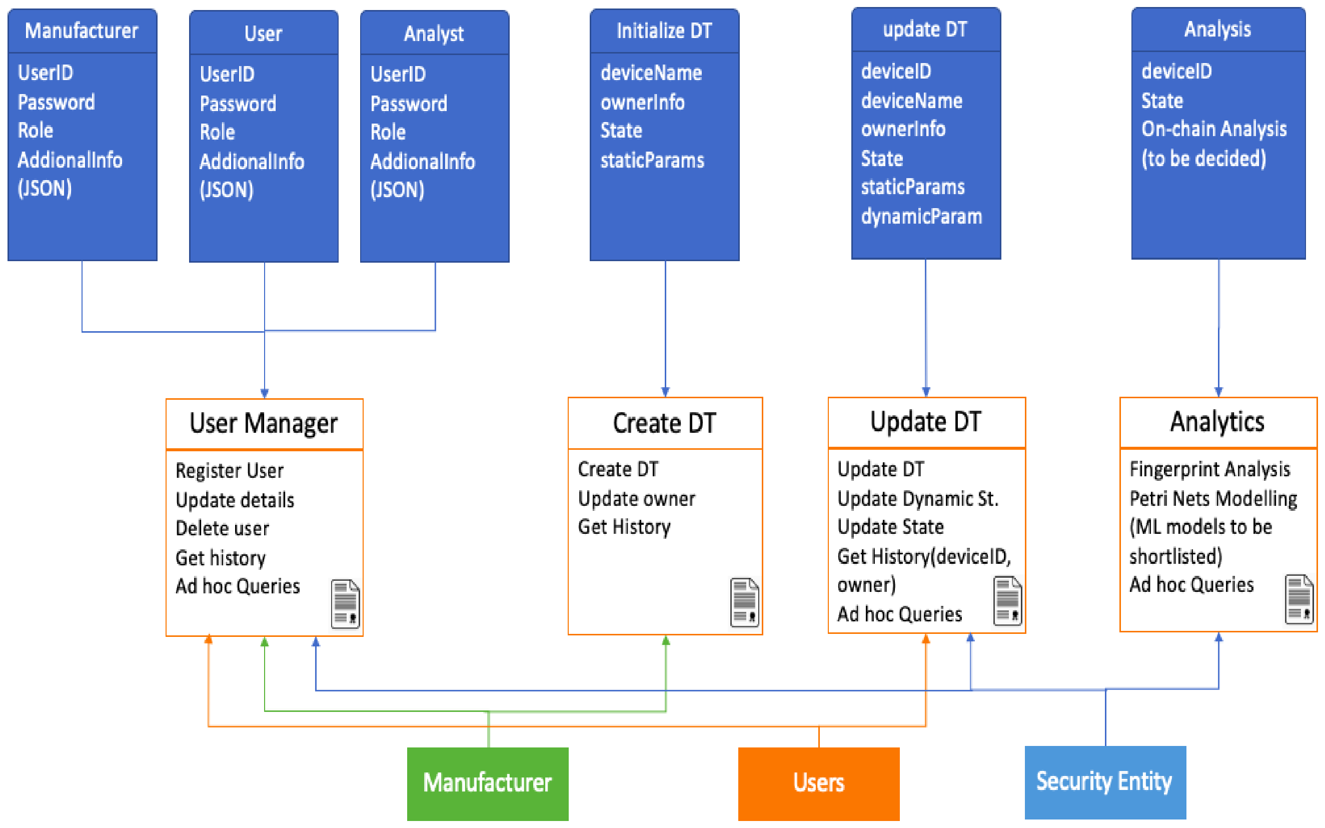
V. Smart Contract Diagram



List of Smart Contracts:

1. User Management:
    a. Entities Registration: Manufacturer/User/Security Entity submits/updates profile info -> Entity receives blockchain access token & Unique ID to access network.
2. DT Create, Sell/Purchase:
    a. Manufacturer creates initial DT of device over network and sets initial price -> User/Owner submits purchase request -> ownership changes.
3. Update Device:
    a. Owner update device with different libraries/Software during the lifecycle -> parallely updating the DT on the network.
4. State Management:
    a. Security Entity (forecasting and analysing) the past (Data from DT) and real-time data from device -> to decide the state of device (Active, Suspend, Unknown)
    b. Owner notified about the status of device -> takes action and updates device accordingly.

VI. Actors and Relationship Diagrams using smart Contracts



Doubts:

1. Can we make Create DT and Update DT as a single contract (having different functions) and return results by just validating the owner?
2. What on chain and off chain analytics we'll be going to perform?
    a. Fingerprints analysis
    b. System calls using word embeddings

VII. **Digital Twin** (considered parameters can be updated according to requirement)

```
{
  "DeviceID": Hash(STATIC_PARAMETERS),
  "DeviceName": "mac",
  "Owner": "Apple",
  "State": enum(active, suspended, deactive, lost),
  "StaticParameter": {
      "timestamp": "",
      "MACaddress": "",
      "SerialNumber": "",
      "Manufacturer": "",
      "Hardware": "Architecture",
      "MemorySize": "xxx",
      "OSInfo": "",
      "StaticIP": "",
      "Price": "$XXX"
    },
  "DynamicParameters":{
        "timestamp": "",
        "LastRestart": "",
        "location": "longitude, latitude",
        "KernelProcess": [
          {
            "ProcessId": "",
            "Priority": "",
            "SharedMemory": "",
            "VirtualMemory": "",
            "CPUusage": ""
          }
        ],
        "IOlatency": "",
        "MessageBuffer": "",
        "KernelBuffer": "",
        "USBbuses": [
            {
              "BusID": "",
              "BusSpeed": "",
              "BusType": ""
            },
          ],
        "PCI":[
            {
              "PCIBusID": "",
              "PCIDevice": ""
            }
          ],
        "SystemActivityReport":[
            "CPUUsage": "",
            "MemoryUtilisation": "",
            "IOdeviceConsumption": "",
            "NetworkStats": "",
            "DiskUsage": "",
            "ProcessAndThreadAllocation": "",
            "BatteryPerformance": "",
            "PlugAndPlay": "",
            "ProcessorPerformance": "",
            "SwapMemory": "",
            "ContextSwitchingStats": "",
            "PagingStats": "",
          ],
        "IPC":{
            "MessageQueue": "",
            "Semaphore": "",
            "SharedMemory": ""
        },
        "NetworkConfiguration": {
          "RoutingTables": "",
          "InterfaceStats": "",
          "NetworkConnections": "",
          "TCPports": "",
          "TCPconnections": "",
          "UDPports": "",
          "UDPconnection": "",
          "ListeningPorts": ""
        },
        "SharedDependencies": {
          "UsedDirectDependencies": "",
          "UnusedDirectDependencies": ""
        },
        "ARPCache": "",
        "ARPTable": "",
        "DNSCache": "",
        "SSLPackets": "",
  }
}
```

VIII. Description of Static Attributes:

MAC Address	Manufacturer Info	OS Info
● Serial Number ● Hardware	● Code ● Model ● Revision ● RAM ● Manufacturer Namel	● Name ● Type ● Version Number ● Architecture ● Version ● Release ● HostName

IX. Description of Dynamic attributes:

1. Kernel Process Information:

Meta Data		
Tasks	MiB Memory (Mebibytes)	MiB Swap
● Total ● Running ● Sleeping ● Stopped ● Zombie	● Total ● Free ● Used ● Buffer/Cache	● Total ● Free ● Used ● Buffer/Cache

- A zombie process or defunct process is a process that has completed execution (via the exit system call) but still has an entry in the process table: it is a process in the "Terminated state".

- A Swap Memory is a space in the Hard Disk of your computer that Operating Systems will use to put the info that is actually on the RAM to free it for another application.

- 1 mebibyte = 2^{20} or 1,048,576 bytes.

Process Information	
PID	**Process Id:** The task's unique process ID, which periodically wraps, though never restarting at zero. In kernel terms, it is a dispatchable entity defined by a task structure.
USER	The owner of the process.

PR	**Priority:** The scheduling priority of the task. If you see `rt' in this field, it means the task is running under real time scheduling priority.
NI	**Nice Value:** A negative nice value means higher priority, whereas a positive nice value means lower priority. Zero in this field simply means priority will not be adjusted in determining a task's dispatch-ability.
VIRT	**Virtual Memory Size (KiB):** The total amount of virtual memory used by the task. It includes all code, data and shared libraries plus pages that have been swapped out and pages that have been mapped but not used.
RES	**Resident Memory Size (KiB):** A subset of the virtual address space (VIRT) representing the non-swapped physical memory a task is currently using. It is also the sum of the RSan, RSfd and RSsh fields. It can include private anonymous pages, private pages mapped to files (including program images and shared libraries) plus shared anonymous pages. All such memory is backed by the swap file represented separately under SWAP. Lastly, this field may also include shared file-backed pages which, when modified, act as a dedicated swap file and thus will never impact SWAP.
SHR	**Shared Memory Size (KiB):** A subset of resident memory (RES) that may be used by other processes. It will include shared anonymous pages and shared file-backed pages. It also includes private pages mapped to files representing program images and shared libraries.
S	**Process Status:** The status of the task which can be one of: ● D = uninterruptible sleep ● I = idle ● R = running ● S = sleeping ● T = stopped by job control signal ● t = stopped by debugger during trace ● Z = zombie
%CPU	**CPU Usage:** The task's share of the elapsed CPU time since the last screen update, expressed as a percentage of total CPU time. In a true SMP environment, if a process is multithreaded and top is not operating in Threads mode, amounts greater than100% may be reported. You toggle Threads mode with the `H' interactive command.  Also for multi-processor environments, if Irix mode is Off, top will operate in Solaris mode where a task's cpu usage will be divided by the total number of CPUs. You toggle Irix/Solaris modes with the `I' interactive command.
%MEM	**Memory Usage (RES)** A task's currently resident share of available physical memory.

TIME+	**CPU Time (hundredths):** The same as TIME, but reflecting more granularity through hundredths of a second.
COMMAND	**Command Name or Command Line:** Display the command line used to start a task or the name of the associated program. You toggle between command line and name with `c', which is both a command-line option and an interactive command.

2. TLS: Transport Layer Security

'Server hello' message	'Client hello' message	Client Key Exchange	‘Server Hello’ Completed
● Agreed Cipher	● Source IP ● Destination IP ● Source Port ● Destination Port	● Public Key	● Certificate Code/ID
	● Version ● Message Length ● Handshake Type ● Handshake Length		

3. DNS: Domain Name System

QD:Query Definition	AN: Answer	NS:Name Server records
● QName: A domain name represented as a sequence of labels, where each label consists of a length octet followed by that number of octets ● QType: ● A two octet code which specifies the type of the query.	● rrName:returns the name of the queried resource ● rData: The data of the response. ● ttl:(time to live) is a setting that tells the DNS resolver how long to cache a query before requesting a new one. ● Type:Two octets containing one of the type codes. This field specifies the meaning of the data in the RDATA field.	● rrName ● rName:Email address of the administrator responsible for this zone. ● mName:This is the name of the primary nameserver for the zone. Secondary servers that maintain duplicates of the zone's DNS records receive updates to the zone from this primary server. ● Type ● Serial:Serial number for this zone. If a secondary name server slaved to this one observes an increase in this number, the slave will assume that the zone has been updated and initiate a zone transfer. ● Retry: The length of time a server should wait for asking an unresponsive primary nameserver for an update again. ● Expire: If a secondary server does not get a response from the primary server for this amount of time, it should stop responding to queries for the zone.

4. IP Addressing:

Device IP	Provides an identity for your device on a network.
Broadcast IP	Used to target all systems on a specific subnet network instead of single hosts.
Netmask	A shorthand for referring to ranges of consecutive IP addresses in the Internet Protocol. They are used for defining networking rules.
GatewayIP	The private IP address assigned to the router.
DNS_information	● DNS IP ● Binary_format: DNS_IP in binary format. ● Name of the DNS Server ● Alias List: A type of DNS record that points the domain name to a hostname instead of an IP address.

5. ICMP: Internet Control Message Protocol

Type , Error	
0,0	Echo reply to ping
3,0	Destination network unreachable
3,1	Destination host unreachable
3,2	Destination protocol unreachable
3,3	Destination port unreachable
3,6	Destination network unknown
3,7	Destination host unknown
4,0	Source quench (unused, TCP provides congestion control)
8,0	Echo request by ping
9,0	Router advertisement (used by RIP)
10,0	Router discovery
11,0	TTL expired (router loop, also used by traceroute)
12,0	IP header bad

6. TCP: Transmission Control Protocol

- Timestamp
- Source Port
- Destination Port

7. System Activity Report
    a. IOStats

tps	Total number of transfers per second that were issued to physical devices. A transfer is an I/O request to a physical device. Multiple logical requests can be combined into a single I/O request to the device. A transfer is of indeterminate size.
rtps	Total number of read requests per second issued to physical devices.
wtps	Total number of write requests per second issued to physical devices.
dtps	Total number of discard requests per second issued to physical devices.
bread/s	Total amount of data read from the devices in blocks per second. Blocks are equivalent to sectors and therefore have a size of 512 bytes.
bwrtn/s	Total amount of data written to devices in blocks per second.
bdscd/s	Total amount of data discarded for devices in blocks per second.

    b. CPU Utilization

%user	Percentage of CPU utilization that occurred while executing at the user level (application). Note that this field includes time spent running virtual processors.
%nice	Percentage of CPU utilization that occurred while executing at the user level with nice priority.
%system	Percentage of CPU utilization that occurred while executing at the system level (kernel). Note that this field includes time spent servicing hardware and software interrupts.
%iowait	Percentage of time that the CPU or CPUs were idle during which the system had an outstanding disk I/O request.
%steal	Percentage of time spent in involuntary wait by the virtual CPU or CPUs while the hypervisor was servicing another virtual processor.

%idle	Percentage of time that the CPU or CPUs were idle and the system did not have an outstanding disk I/O request.

c. Processor Queue

runq-sz	Run queue length (number of tasks running or waiting for run time).
plist-sz	Number of tasks in the task list.
ldavg-1	System load average for the last minute. The load average is calculated as the average number of runnable or running tasks (R state), and the number of tasks in uninterruptible sleep (D state) over the specified interval.
ldavg-5	System load average for the past 5 minutes.
ldavg-15	System load average for the past 15 minutes.
blocked	Number of tasks currently blocked, waiting for I/O to complete.

d. Memory Utilization

kbmemfree	Amount of free memory available in kilobytes.
kbavail	Estimate of how much memory in kilobytes is available for starting new applications, without swapping. The estimate takes into account that the system needs some page cache to function well, and that not all reclaimable slab will be reclaimable, due to items being in use. The impact of those factors will vary from system to system.
kbmemused	Amount of used memory in kilobytes. This value does not take into account memory used by the kernel itself.
%memused	Percentage of used memory.
kbbuffers	Amount of memory used as buffers by the kernel in kilobytes.
kbcached	Amount of memory used to cache data by the kernel in kilobytes.
kbcommit	Amount of memory in kilobytes needed for current workload. This value is an estimate of how much RAM/swap is needed to guarantee that there never is out of memory.
%commit	Percentage of memory needed for current workload related to the total amount of memory (RAM+swap). This number may be greater than 100% because the kernel usually over commits memory.

kbactive	Amount of active memory in kilobytes (memory that is used more recently and usually not reclaimed unless absolutely necessary).
kbinact	Amount of inactive memory in kilobytes (memory which is less recently used. It is more eligible to be reclaimed for other purposes).
kbdirty	Amount of memory in kilobytes waiting to get written back to the disk.

e. Disk Device Status

tps	Total number of transfers per second that were issued to physical devices. A transfer is an I/O request to a physical device. Multiple logical requests can be combined into a single I/O request to the device. A transfer is of indeterminate size.
rkB/s	Number of kilobytes read from the device per second.
wkB/s	Number of kilobytes written to the device per second.
dkB/s	Number of kilobytes discarded for the device per second.
areq-sz	The average size (in kilobytes) of the I/O requests that were issued to the device.
aqu-sz	The average queue length of the requests that were issued to the device.
await	The average time (in milliseconds) for I/O requests issued to the device to be served. This includes the time spent by the requests in queue and the time spent servicing them.
%util	Percentage of elapsed time during which I/O requests were issued to the device (bandwidth utilization for the device). Device saturation occurs when this value is close to 100% for devices serving requests serially. But for devices serving requests in parallel, such as RAID arrays and modern SSDs, this number does not reflect their performance limits.