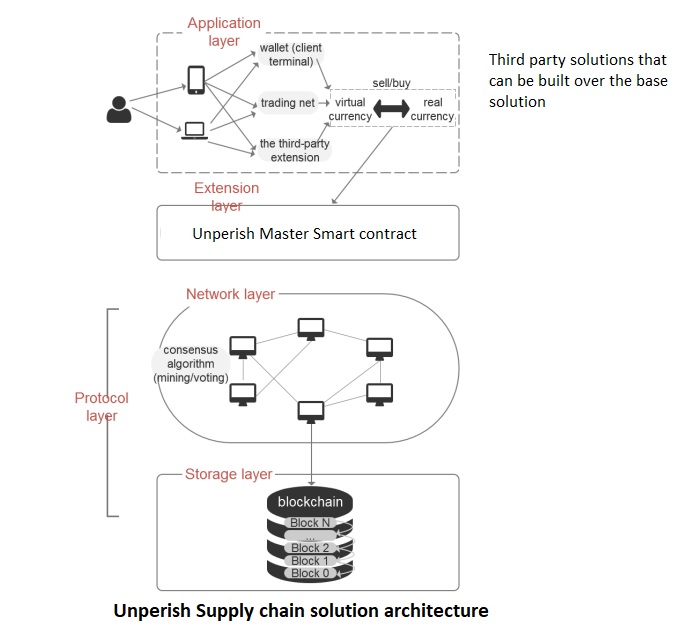
Architecture of Unperish Supply chain

Solution goal: A Blockchain solution for solving supply chain pain points of Covid 19 vaccine, this solution enables:

1. transparent tracking of all raw material, intermediate products and final finished goods throughout the supply chain,
2. building implicit trust between interacting parties since all records are un-mutable once recorded
3. reducing entry and exit barriers for participating parties

Solution Architecture diagram



Store ERC721 tokens on Ethereum blockchain

Component Technologies

1. IoT devices/sensors:

IoT Devices and sensors will be used to capture the real world statistics of the vaccine through out the supply chain. Technologies like RFIDs, motion sensors, GPS trackers, temperature sensors and gyroscopic sensors will be used to gather storage and movement information

1. Ethereum Blockchain:

ERC721 tokens will represent vaccine on the blockchain. All movements will be tracked as transactions on the chain and will be accessible by all to view and verify.

Smart contract of supply chain will control the Token creation and creating appropriate transactions on the blockchain as the vaccine completes its journey through the supply chain.

1. Webpage front end:

Basic webapp which interfaces with the blockchain. Distributors can check the status of the vaccines on the blockchain.

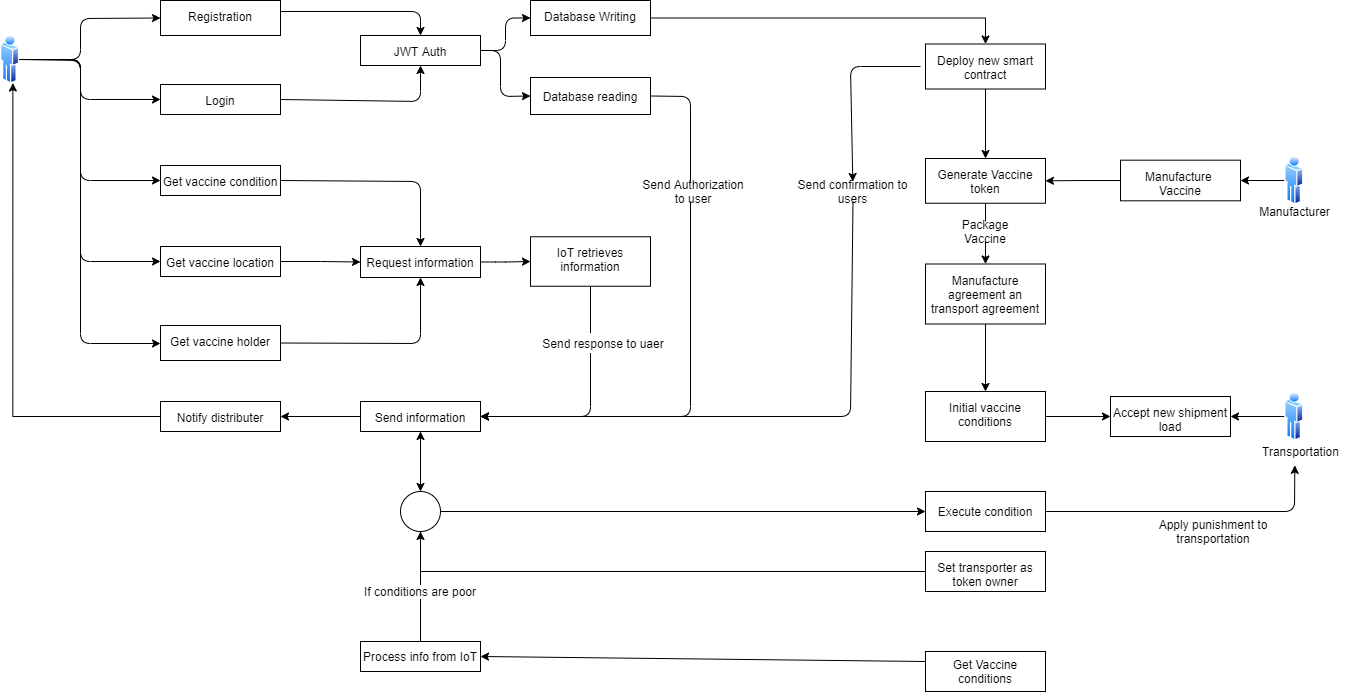
1. Middleware:

Rest APIs will be created and exposed to interact with the blockchain so that all future third party integrations are seamless.

1. Backend:

Database is used to store other relevant information of the supplychain transactions for tracking. This is done to reduce costs of storing data on Ethereum blockchain.

UnPerish Flow



Data Definition

1. Smart Contract
   1. Data Storage:

Vaccine struct

uint256 id

string manufacturer

bytes6 serialno

int16 thermal

bytes26 location (Byte array to store latitude and longitude in the form

<lat>;<long> Example: -89.12345678;-168.12345678)

address requestingDistributor

mapping(uint256 => Vaccine) private vaccines (Maps token ids to vaccine info)

uint256 private nextId (Next token id to be minted)

* 1. Funtions
     1. addManufacturer: Add address as manufacturer role

Inputs: (address manufacturer) public onlyAdmin()

* + 1. addTransporter: Add address as transporter role

Inputs: (address transporter) public onlyAdmin()

* + 1. addDistributor: Add address as distributor role

Inputs: (address distributor) public onlyAdmin()

* + 1. makeVaccine: Mints a new Vaccine token

Inputs: (string memory \_manufacturer, bytes6 \_serialno, int16 \_thermal, bytes26 \_location) public onlyManufacturer()

* + 1. orderVaccine: Set requesting Distributor in Vaccine token

Inputs: (uint256 \_id) public onlyDistributor()

* + 1. thermalMonitor: Update current thermal and location info

Inputs: (uint256 \_id, int16 \_thermal, bytes26 \_location) public onlyTransporter() onlyOwner(\_id)

* + 1. transferVaccine: Transfer Vaccine

Inputs: (uint256 \_id, address \_to, int16, \_thermal, bytes26 \_location) public onlyOwner(\_id)

* + 1. getVaccineTrackingInfo: Get current vaccine info. Returns thermal, location and current owner

Inputs: (uint256 \_id) public view onlyRequestingDistributor(\_id)

returns (int16, bytes26, address)

1. Database:

name (string)

password (string)

address(string)

registration date(date)

Backend

**1. Endpoints**

ADMINISTRATOR

|  |  |
| --- | --- |
| **EndPoint** | **Description** |
| /admin/addManufacturer | This endpoint calls addManufacturerfrom smart contract |
| /admin/addTransporter | This endpoint calls addTransporterfrom smart contract |
| /admin/addDistributor | This endpoint calls addDistributorfrom smart contract |

DISTRIBUTOR

|  |  |
| --- | --- |
| **EndPoint** | **Description** |
| /distributors/index | get list of registered distributors in db |
| /distributors/new | register new distributor |
| /distributors/login | distributor login |
| /distibutors/vaccineInfo/:tokenId | Get current info of vaccine. This endpoint callsgetVaccineTrackingInfo from smart contract |
| /distributors/orderVaccine/:tokenId | This endpoint calls orderVaccine from smart contract |

MANUFACTURER

|  |  |
| --- | --- |
| **EndPoint** | **Description** |
| /manufacturers/transferVaccine | This endpoint calls transferVaccine from smart contract |
| /manufacturers/makeVaccine | This endpoint calls makeVaccine smart contract |

TRANSPORTER

|  |  |
| --- | --- |
| **EndPoint** | **Description** |
| /transporters/transferVaccine | This endpoint calls transferVaccine from smart contract |

ORACLE (IoT)

|  |  |
| --- | --- |
| **EndPoint** | **Description** |
| /transporters/thermalMonitor | This endpoint calls thermalMonitor from smart contract |

Tech Stack

Smart Contract

* [Solidity](https://solidity.readthedocs.io/) - smart contract programming language
* [Truffle](https://www.trufflesuite.com/) - dApp environment
* [Ethereumjs-util](https://www.npmjs.com/package/ethereumjs-util) - utility functions for Ethereum
* [Truffle-assertions](https://www.npmjs.com/package/truffle-assertions) - additional assertions for truffle
* [Bignumber.js](https://www.npmjs.com/package/bignumber.js) - library to handle big numbers

Backend

* [Express.js](http://expressjs.com/) - web application framework
* [MongoDB](https://www.mongodb.com/) - NoSQL database
* [Mongoose](https://mongoosejs.com/) - object data modeling (ODM) library for MongoDB and Node.js
* [Async](https://caolan.github.io/async/v3/) - library to perform asynchronous operations
* [Express validator](https://express-validator.github.io/docs/) - middleware to validate data
* [Bcryptjs](https://www.npmjs.com/package/bcryptjs) - library to perform cryptography
* [JWT.IO](https://jwt.io/) - JSON Web Tokens to allow, decode, verify, and generate JWT
* [Jest](https://jestjs.io/) - library for tests
* [Web3.js](https://web3js.readthedocs.io/) - interact with smart contracts
* [Dotenv](https://www.npmjs.com/package/dotenv) - loads environment variables from a .env file

## Frontend

* [Rimble](https://rimble.consensys.design/) - design system
* [ReactJS](https://reactjs.org/) - frontend library
* [Axios](https://www.npmjs.com/package/axios) - HTTP requests