National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Shanmugasundar Ramesh		
Student ID:	X22171649		
Programme:	Masters in Cybersecurity	**Year:**	2023
Module:	Msc Research Project		
Supervisor:	Michael Pantridge		
Submission Due Date:	14/12/2023,31/01/2024		
Project Title:	ENHANCING SECURE COMMUNICATIONS PROTOCOL IN ADAS (AUTONOMOUS DRIVING SYSTEM) SYSTEM		
Word Count:	1149 **Page Count 10**		

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Shanmugasundar Ramesh
Date:	31/01/2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Shanmugasundar Ramesh

Student ID: x22171649

1.Introduction

The web application which is deployed is constructed using React.js which is the platform to show the proof of work for the **novel secure communication protocol** for the ADAS system which involves the combination of **MIDM-AES-GCM** with **BLOCKCHAIN.** The evaluation of the proof of work is done using GANACHE (a platform for testing etherium block chain concept) this is used to deploy smart contracts in the block chain which is coded in solidity language .These smart contracts are essential because they automate the actions that should be executed otherwise be completed by the parties in the agreement so here we deploying 4 smart contracts. The connection is established between the web application and block chain through a platform called Metamask . when the data gets encrypted in the start using the encryption method MIDM-AES-GCM we get as the cipher text this cipher text is then pushed into the etherium block chain (Ganache) through metamask by paying the gas price. So now we can see the block created in the Ganache platform . so during the decryption process the data gets fetched from the block chain and then it gets decrypted using the MIDM-AES-GCM process and then with help of smart contracts the data is given back.

2.Hardware Requirements

Processor (CPU) : Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz, 1992 Mhz, 4 Core(s), 8 Logical Processor(s)

Memory (RAM) : 32gb RAM/4gb RAM(minimum)

3. Software Requirements

- Operating system : Windows 10 /Windows 8 (minimum)
- React.js: The web application interface is built on react.js which act as base for the data to be fetched
- Javascript libraries :This the part where it brings the cryptographic function of MIDM-AES-GCM
- Yarn(version 1.22.21): This needs to be installed to manage all the javascript project dependencies
- Npm and node.js : This is installed to fetch the java libraries(which has the encryption part AES-GCM-MIDM) and the project dependencies .
- Ganache platform : This can be downloaded from the truffle suite
- Solidity Contract Deployment: compile the solidity code for the smart contract and deploy it
- Metamask : If you are using the chrome browser u can install the Metamask extension from the chrome extension (Metamask chrome extension) .

4.Implementation

First we deploy the smart contracts using truffle for that we install NPM and then we run the run the command “truffle init” his will generate a project structure with the following directories: contracts/ for Solidity contracts, migrations/ for JavaScript files helping deploy contracts, and test/ for test scripts then we compile the smart contract that is coded in solidity language then we migrate the data using the command truffle migrate which deploys the smart contracts . so in the below fig 1 ,fig 2,fig 3,fig 4 . we have deployed four smart contract namely Sensordata ,Auth, migration and vehicle data . so now it goes into the block chain of ethereum.

```
3_sensordata_migrations.js
==========================

   Replacing 'SensorData'
   ----------------------
   > transaction hash:    0xc017bf7b202958058960e8648b215adcb9896e6a3a67d3e8091c25107bbb371c
   > Blocks: 0            Seconds: 0
   > contract address:    0xcd2611DA66Bcd8485EEDc0798BFfEAe58C455c3A
   > block number:        5
   > block timestamp:     1701350852
   > account:             0xe13a3C603F0450fa9b2eD20E73B7d113eBd7Eee3
   > balance:             99.995382198972027932
   > gas used:            597565 (0x91e3d)
   > gas price:           3.031984417 gwei
   > value sent:          0 ETH
   > total cost:          0.001811807768144605 ETH

   > Saving migration to chain.
   > Saving artifacts
   -------------------------------------
   > Total cost:     0.001811807768144605 ETH
```

Fig 1

```
2_deploy_migration.js
=====================

   Replacing 'Auth'
   ----------------
   > transaction hash:    0x27105b8bb98fd1184241bbcb5efae7a92190fcc17a77007d0a910a31390f91d7
   > Blocks: 0            Seconds: 0
   > contract address:    0x2f3E162046CeEf2435F612C146f8Fcd425Ab26F9
   > block number:        3
   > block timestamp:     1701350852
   > account:             0xe13a3C603F0450fa9b2eD20E73B7d113eBd7Eee3
   > balance:             99.997283535603534045
   > gas used:            541752 (0x84438)
   > gas price:           3.178366198 gwei
   > value sent:          0 ETH
   > total cost:          0.001721886244498896 ETH

   > Saving migration to chain.
   > Saving artifacts
   -------------------------------------
   > Total cost:     0.001721886244498896 ETH
```

Fig 2

```
Starting migrations...
======================
> Network name:    'development'
> Network id:      5777
> Block gas limit: 6721975 (0x6691b7)


1_initial_migration.js
======================

   Replacing 'Migrations'
   ----------------------
   > transaction hash:    0x63051a2235ccaceb6fa379a87685f6d338a2651ee2968eab1d5955ff2112eed6
   > Blocks: 0            Seconds: 0
   > contract address:    0x9E3a4b794dF20ebd0C40a6A8E5da31D60cbB801A
   > block number:        1
   > block timestamp:     1701350851
   > account:             0xe13a3C603F0450fa9b2eD20E73B7d113eBd7Eee3
   > balance:             99.99915573025
   > gas used:            250154 (0x3d12a)
   > gas price:           3.375 gwei
   > value sent:          0 ETH
   > total cost:          0.00084426975 ETH

   > Saving migration to chain.
   > Saving artifacts
   -------------------------------------
   > Total cost:       0.00084426975 ETH
```

Fig 3

```
4_vehicle_migrations.js
=======================

   Replacing 'VehicleData'
   -----------------------
   > transaction hash:    0x1ab983bd6acde40741e1f6f30563c593841e8f61e41f216d7c583f546c0f4cdf
   > Blocks: 0            Seconds: 0
   > contract address:    0xEf825563aaa37d84C8E30A3166445694c54afEe1
   > block number:        7
   > block timestamp:     1701350852
   > account:             0xe13a3C603F0450fa9b2eD20E73B7d113eBd7Eee3
   > balance:             99.993552625169544023
   > gas used:            597565 (0x91e3d)
   > gas price:           2.918157168 gwei
   > value sent:          0 ETH
   > total cost:          0.00174378858809592 ETH

   > Saving migration to chain.
   > Saving artifacts
   -------------------------------------
   > Total cost:     0.00174378858809592 ETH

Summary
=======
> Total deployments:   4
> Final cost:          0.006121752350739421 ETH
```

Fig 4

Setting up the ganache and Metamask :

So in ganache we different Accounts comprising of ethereum which contains a Address and a private key(this is illustratively shown in fig 5 and fig 6) so we install the metamask and create a account and add the crypto wallet of the the account from ganache to metamask by pasting the private key into metamask where now metamask is the bridge between the browser and the ganache platform (ethereum cryptowallet) this is illustratively shown in fig 7 and fig 8 so what we have done is created an account named “account 1” and transferred all the etherium to metamask which act as bridge .

Fig 5

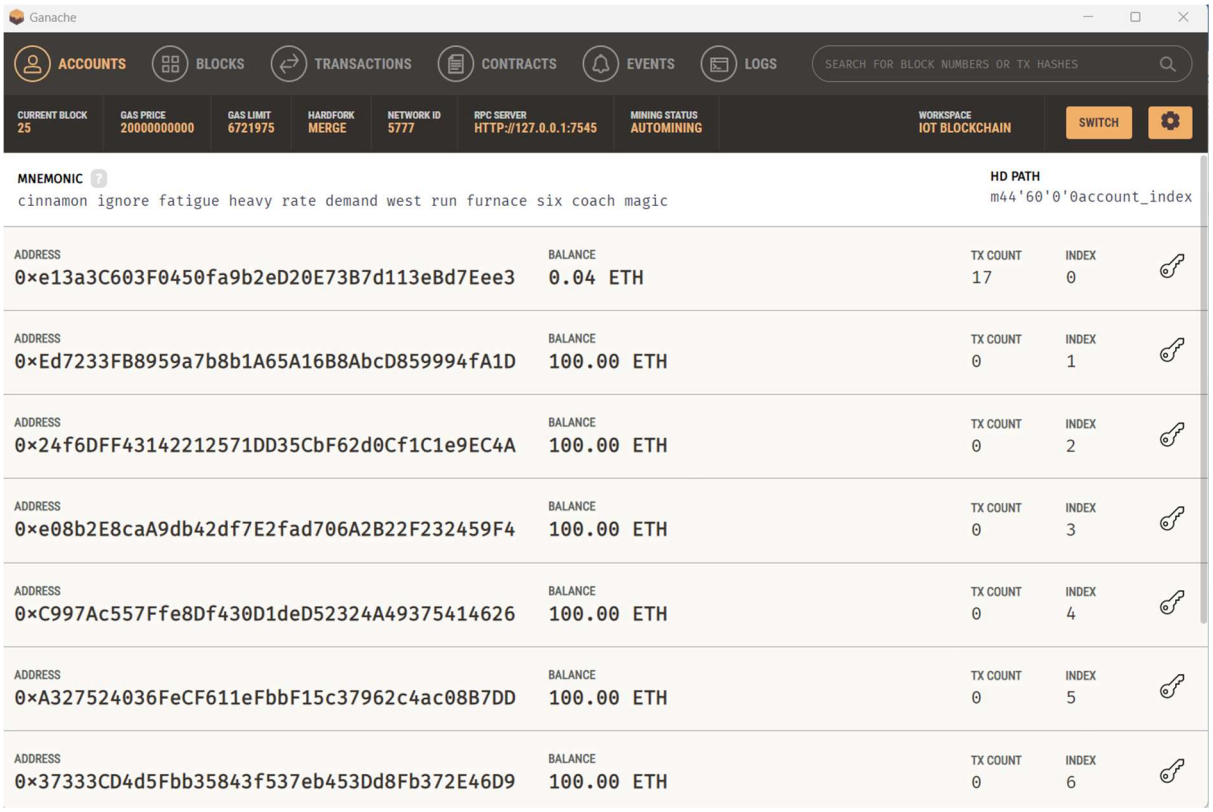
Ganache
ACCOUNTS
BLOCKS
TRANSACTIONS
CONTRACTS
EVENTS
LOGS
SEARCH FOR BLOCK NUMBERS OR TX HASHES
CURRENT BLOCK 25
GAS PRICE 20000000000
GAS LIMIT 6721975
HARDFORK MERGE
NETWORK ID 5777
RPC SERVER HTTP://127.0.0.1:7545
MINING STATUS AUTOMINING
WORKSPACE IOT BLOCKCHAIN
SWITCH
MNEMONIC
cinnamon ignore fatigue heavy rate demand west run furnace six coach magic
HD PATH
m44'60'0'0account_index
ADDRESS 0×e13a3C603F0450fa9b2eD20E73B7d113eBd7Eee3 BALANCE 0.04 ETH TX COUNT 17 INDEX 0
ADDRESS 0×Ed7233FB8959a7b8b1A65A16B8AbcD859994fA1D BALANCE 100.00 ETH TX COUNT 0 INDEX 1
ADDRESS 0×24f6DFF43142212571DD35CbF62d0Cf1C1e9EC4A BALANCE 100.00 ETH TX COUNT 0 INDEX 2
ADDRESS 0×e08b2E8caA9db42df7E2fad706A2B22F232459F4 BALANCE 100.00 ETH TX COUNT 0 INDEX 3
ADDRESS 0×C997Ac557Ffe8Df430D1deD52324A49375414626 BALANCE 100.00 ETH TX COUNT 0 INDEX 4
ADDRESS 0×A327524036FeCF611eFbbF15c37962c4ac08B7DD BALANCE 100.00 ETH TX COUNT 0 INDEX 5
ADDRESS 0×37333CD4d5Fbb35843f537eb453Dd8Fb372E46D9 BALANCE 100.00 ETH TX COUNT 0 INDEX 6

Fig 6

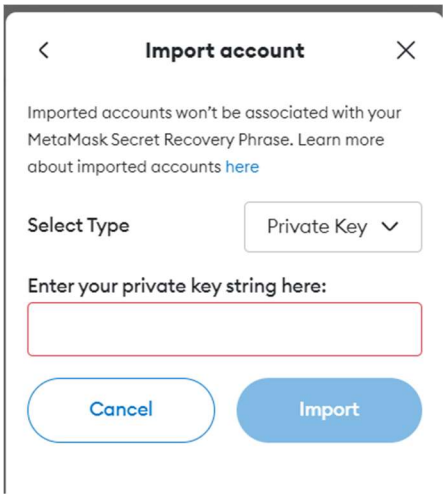
Import account
Imported accounts won't be associated with your MetaMask Secret Recovery Phrase. Learn more about imported accounts here
Select Type
Private Key
Enter your private key string here:
Cancel
Import

Fig 7

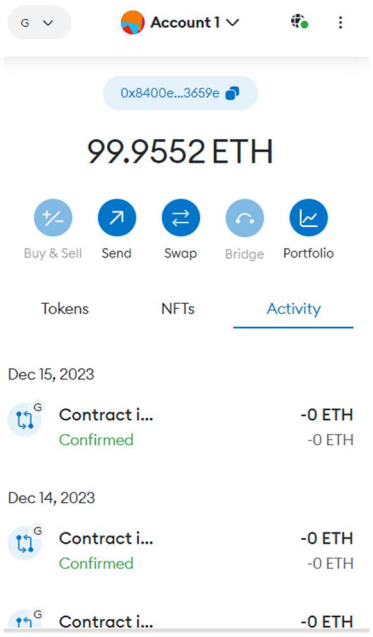

Fig 8

Setting up the web application :

We go the directory of the application and then we open a terminal in the directory (*if your using visual studio go to the directory open the terminal) so then we start yarn to deploy the web application. So the fig 9 depicts the terminal where the we started yarn and the application is being deployed on port 3000 in fig 10

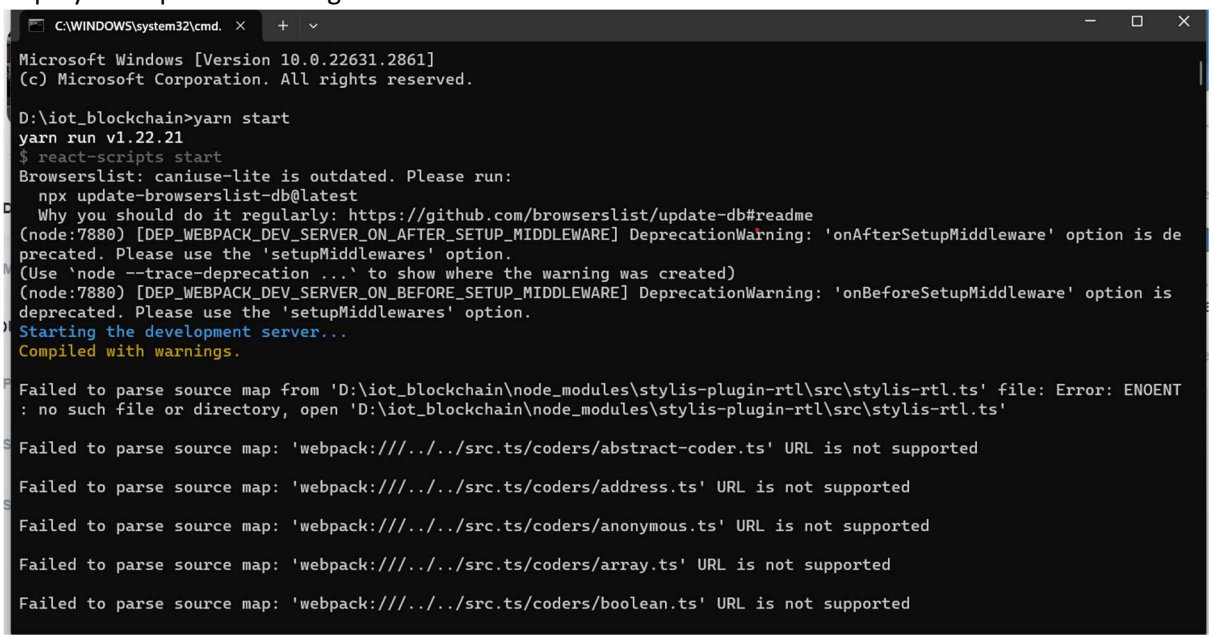

Fig 9

After the web application deployed we will have page as shown in Fig 10 where we have the application which act as the ADAS system containing all the sensor data and moreover to show the proof of work we have developed in a such a there is a button for encryption and decryption of the data .

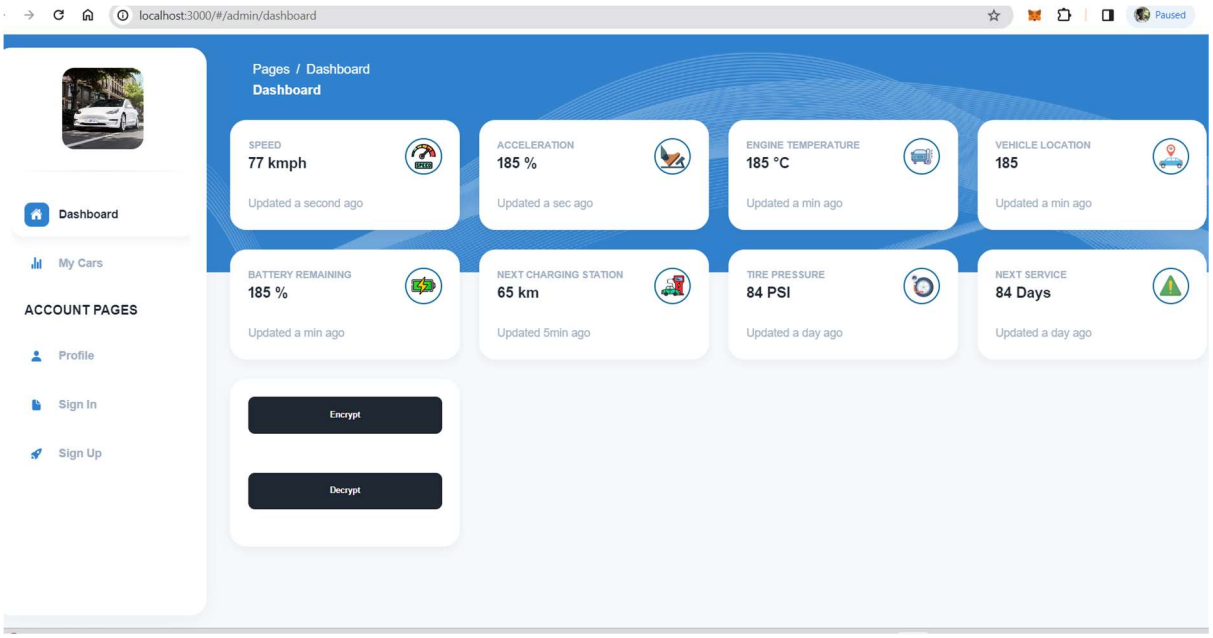

Fig 10

Working of the encryption:

So the working is very simple now The data gets fetched from the sensor (in real time) but now the data has been fetched from the application then we click on encrypt the data gets encrypted with AES-GCM in fig 11 and then pushes the data using Meta mask to the etherium block chain so the metamask pops up asking to pay the gas price of pushing the cipher text into the block chain (etherium) fig 12 then we have the proof that the data is created in the block chain through ganache we we click the block tab we see the data is created as per fig 13

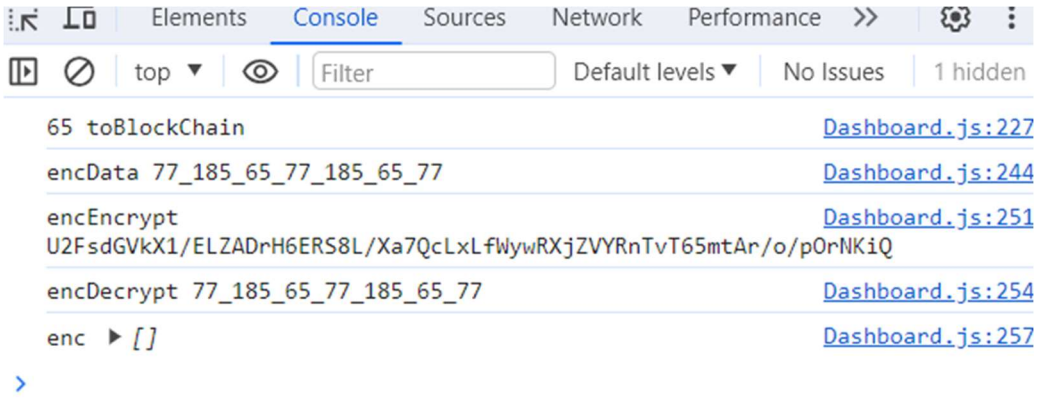

Fig 11

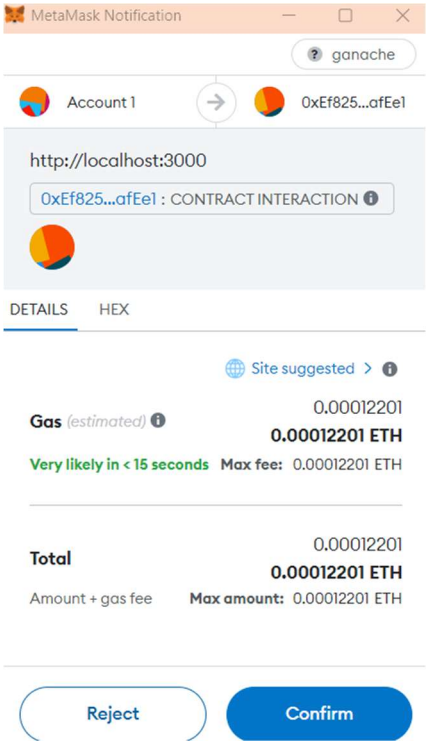
MetaMask Notification
ganache
Account 1
0xEf825...afEe1
http://localhost:3000
0xEf825...afEe1 : CONTRACT INTERACTION
DETAILS
HEX
Site suggested
Gas (estimated)
0.00012201
0.00012201 ETH
Very likely in < 15 seconds
Max fee: 0.00012201 ETH
Total
0.00012201
0.00012201 ETH
Amount + gas fee
Max amount: 0.00012201 ETH
Reject
Confirm

Fig 12

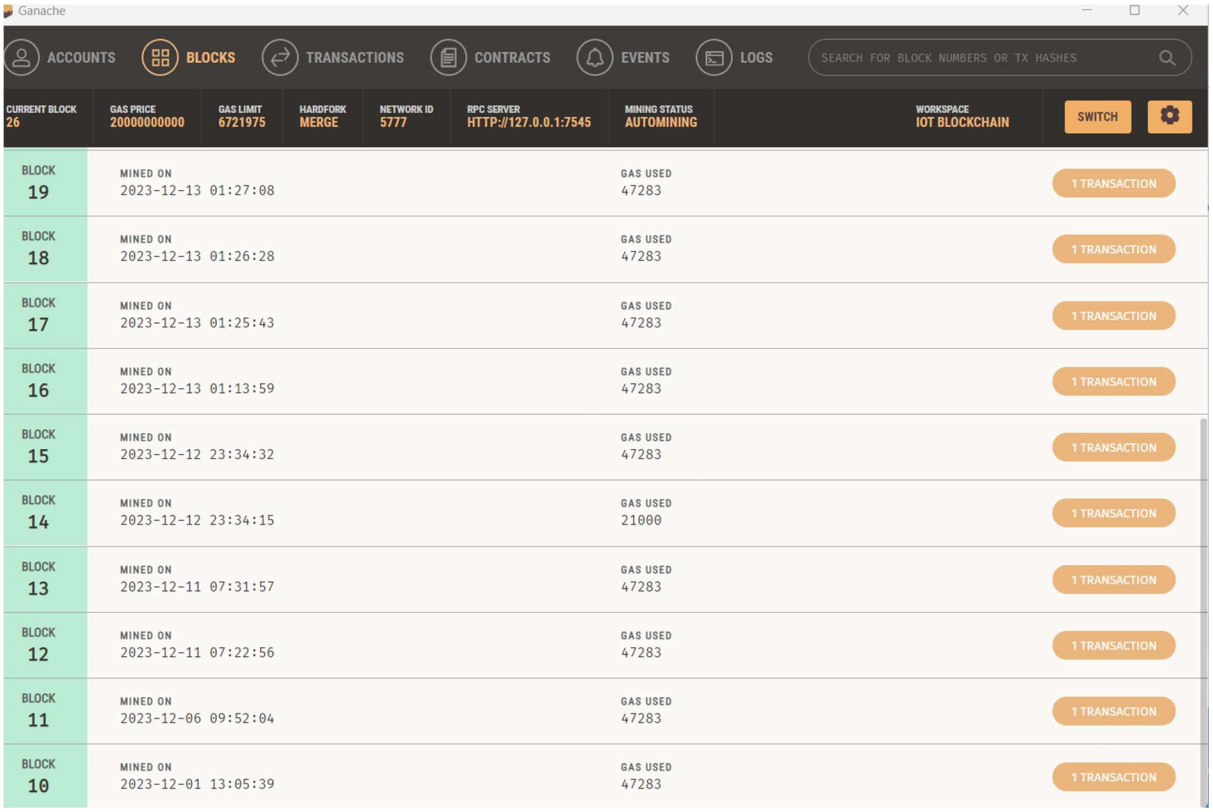
Ganache
ACCOUNTS
BLOCKS
TRANSACTIONS
CONTRACTS
EVENTS
LOGS
SEARCH FOR BLOCK NUMBERS OR TX HASHES
CURRENT BLOCK 26
GAS PRICE 20000000000
GAS LIMIT 6721975
HARDFORK MERGE
NETWORK ID 5777
RPC SERVER HTTP://127.0.0.1:7545
MINING STATUS AUTOMINING
WORKSPACE IOT BLOCKCHAIN
SWITCH
BLOCK 19 MINED ON 2023-12-13 01:27:08 GAS USED 47283 1 TRANSACTION
BLOCK 18 MINED ON 2023-12-13 01:26:28 GAS USED 47283 1 TRANSACTION
BLOCK 17 MINED ON 2023-12-13 01:25:43 GAS USED 47283 1 TRANSACTION
BLOCK 16 MINED ON 2023-12-13 01:13:59 GAS USED 47283 1 TRANSACTION
BLOCK 15 MINED ON 2023-12-12 23:34:32 GAS USED 47283 1 TRANSACTION
BLOCK 14 MINED ON 2023-12-12 23:34:15 GAS USED 21000 1 TRANSACTION
BLOCK 13 MINED ON 2023-12-11 07:31:57 GAS USED 47283 1 TRANSACTION
BLOCK 12 MINED ON 2023-12-11 07:22:56 GAS USED 47283 1 TRANSACTION
BLOCK 11 MINED ON 2023-12-06 09:52:04 GAS USED 47283 1 TRANSACTION
BLOCK 10 MINED ON 2023-12-01 13:05:39 GAS USED 47283 1 TRANSACTION

Fig 13

Decryption part:

So when decrypt the data the data gets fetched from the particular block and the decrypts the AES-GCM cipher then the sensor data goes through the ADAS system to pass it on through the respective sensors to executed as shown In fig 14

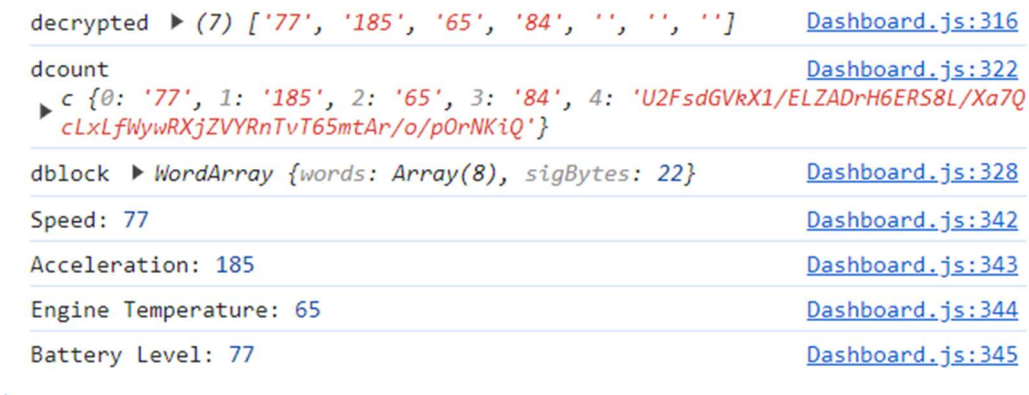

Fig 14