IBM Blockchain Platform Hands-On

Blockchain Concepts

Table of Contents

Disclaimer

1 Overview of the lab environment and scenario

This lab allows you to experience a blockchain solution from the point of view of a set of end-users, and in doing so learn about key blockchain concepts. It is not meant to be a technical introduction to blockchain but will instead focus on the properties of the business network and value that blockchain brings.

The use-case we will work through is one that demonstrates the **lifecycle of a new car**, from the manufacturing and purchasing through to delivery and insurance. It is a good blockchain use-case because there is a defined business network and an identifiable need for trust between the participants of the network.

In this lab, you will be playing the role of the four personas who use the vehicle lifecycle system:

- *Paul* - the buyer/owner of a car
- *Mike* - an employee for the car manufacturer ("Arium")
- *Debbie* - an administrator for the regulator ("Vehicle & Drivers Authority" or "VDA")
- *Tommen* - an Insurer from an insurance company called Prince

These personas together work on ordering, building, transferring ownership of a vehicle while keeping all the other parties in the network updated and building the trust between them to allow them to work together efficiently.

In this lab, each user's application will be represented by a separate tab in our web browser; of course, in a real blockchain network they each user will be running on different systems in different locations, although all connecting in to the same (but distributed) blockchain network.

The lab environment consists of:

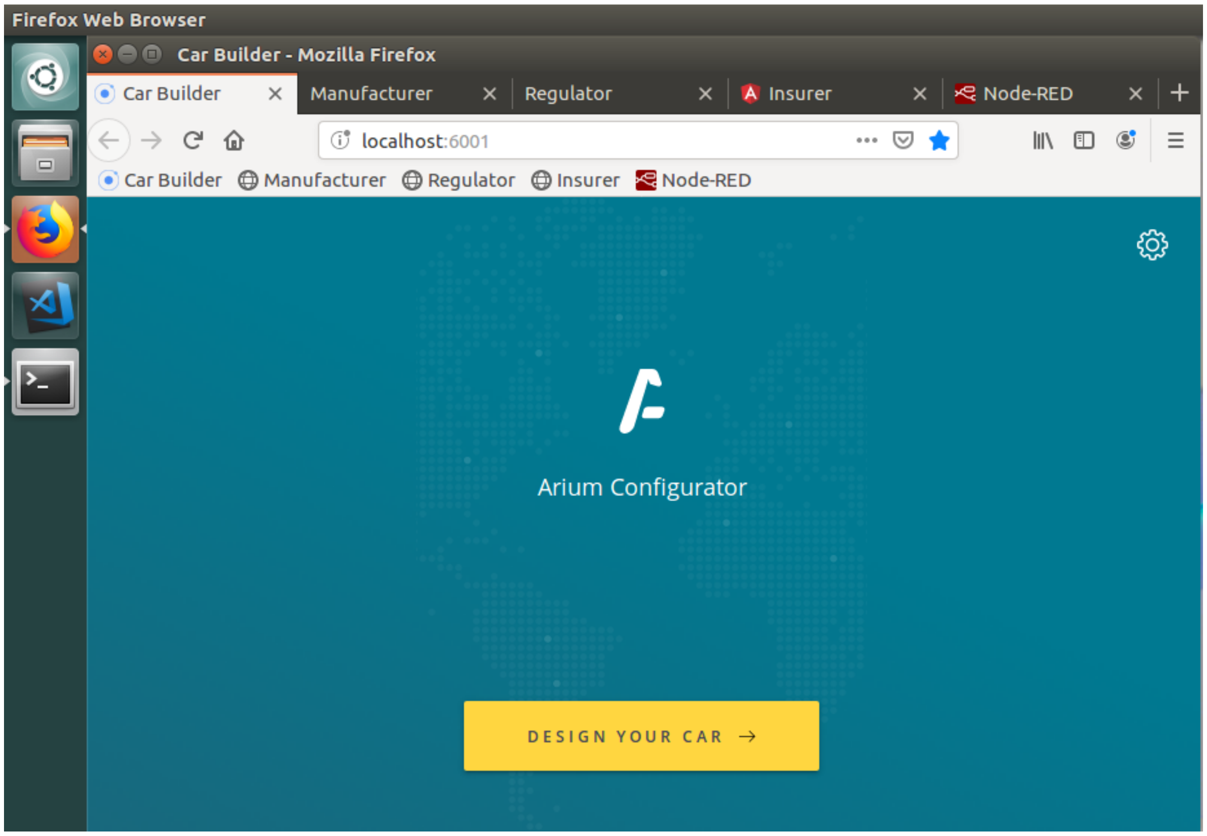
- Ubuntu 16.04
- Hyperledger Fabric V1.4.6
- Vehicle Manufacture demo: https://github.com/IBM-Blockchain/vehicle-manufacture/tree/v1.1.1

Start here. Instructions are always shown on numbered lines like this one:

__ **1.** If it is not already running, start the virtual machine for the lab. Your instructor will tell you how to do this if you are unsure.

__ **2.** Wait for the image to boot and for the associated services to start. This happens automatically but might take several minutes. The image is ready to use when the web browser is visible and five tabs fully loaded, as per the screenshot below.

Note: If it asks you to login, the userid and password are both "**blockchain**".



1.1 Lab Scenario

While the virtual machine is starting, let's recap a few blockchain concepts and introduce the scenario.

The most generally accepted definition of a blockchain is of a ***shared, replicated ledger***.

Ledgers have been around for hundreds of years and are records of what a business has done. They're important systems of record because they describe a business's inputs and outputs and thereby give an indication of its worth. Essentially, they are a log of transactions – a transaction being a change in state of an asset.

The problem with ledgers is that each one is owned by a single business, which means that when one business transacts with another, ledgers can get out of sync. What happens when the transaction I've recorded on my ledger doesn't tally with the transaction you've recorded on your ledger? Disputes inevitably occur which need to be resolved through a reconciliation process. This can be slow and expensive.

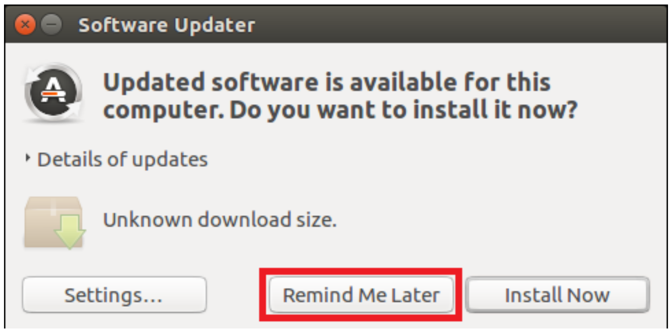
By having a shared ledger, it means that all participants of the business network see the same ledger. By replicating it across the business network, it means that the ledger is not held in any one single place, which would otherwise make it vulnerable to outages and malicious attack.

Consider the business network that surrounds the purchase and ownership of a car. Today, each participant (for example, manufacturer, insurer or regulator) has their own ledger and the processes that allow them to interact with each other varies from company to company and can be time consuming to complete. Connectivity between participants is typically done point-to-point using a variety of processes – some manual or slow (e.g. sending a letter), and some automated (e.g. file, REST API, B2B messaging). This plethora of processes is expensive to maintain and can be vulnerable to attack.

In this scenario we will replace these disparate ledgers with a single blockchain, and the individual business processes with a single shared one. By doing this we will make the overall process much quicker and less prone to error.

We will experience the solution through the eyes of four key members of the business network: a private purchaser, the manufacturer, the regulator and the insurer. We will start by looking at the ordering process, as experienced by the buyer.

Note*: if you get an "Software Updater" pop-up at any point during the lab, please click "****Remind Me Later****":*


Software Updater

Updated software is available for this computer. Do you want to install it now?

Details of updates

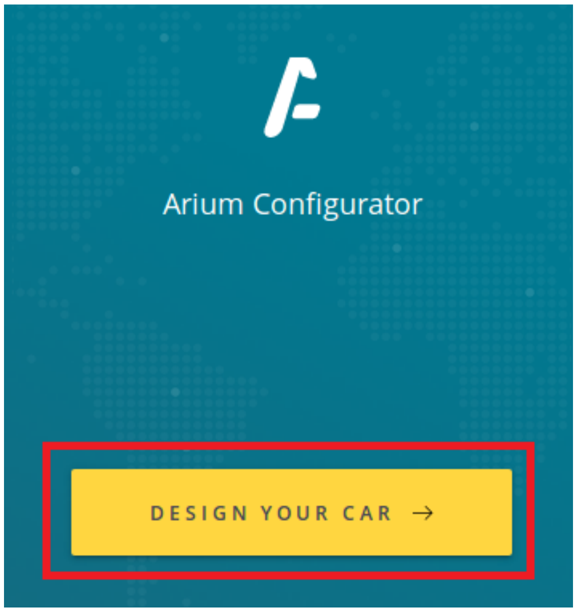
Unknown download size.

2 Running the lab

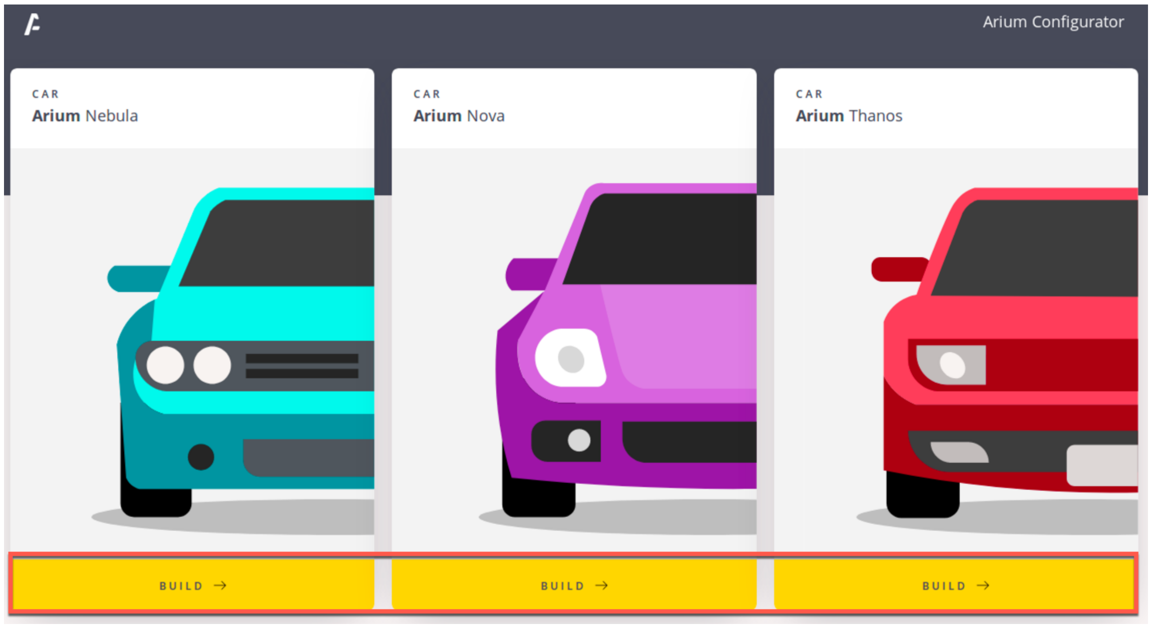
2.1 Ordering the car

__ **3.** If it is not already selected, switch to the "**Car Builder**" tab in the web browser (running at localhost:6001), which is the app through which you as Paul will design and order your new car.

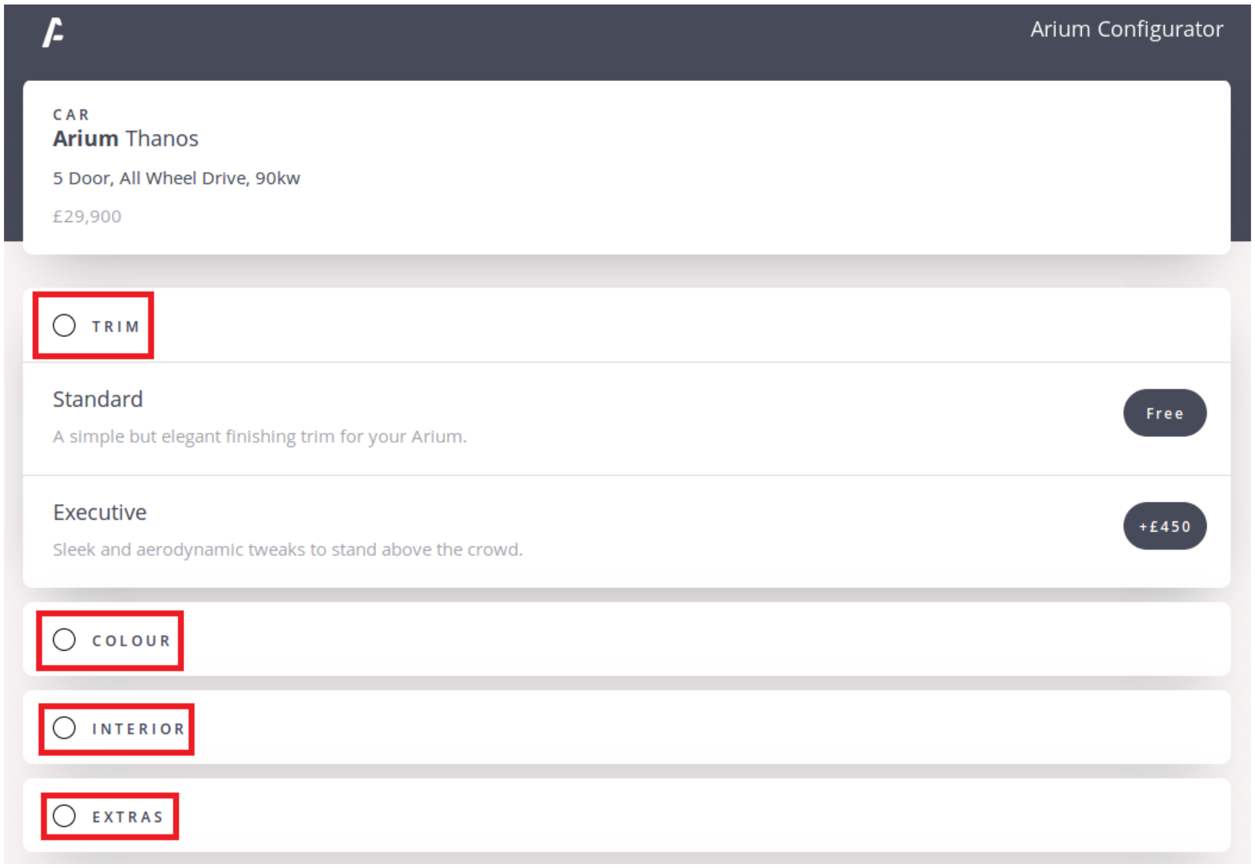
__ **4.** In the **Arium Configurator**, click **'Design Your Car'**.



__ **5.** From the configurator, choose one of the three cars you like and click on **Build** for the car of your choice:



__ **6.** Fill in the different options for your choice of car:



Arium Configurator

CAR
Arium Thanos
5 Door, All Wheel Drive, 90kw
£29,900

TRIM

Standard — A simple but elegant finishing trim for your Arium. — Free

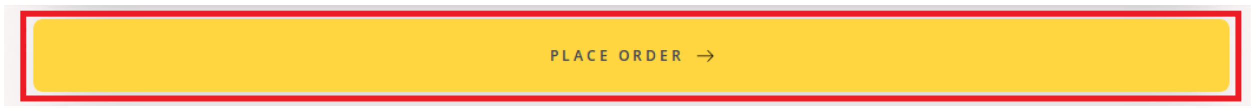
Executive — Sleek and aerodynamic tweaks to stand above the crowd. — +£450

COLOUR

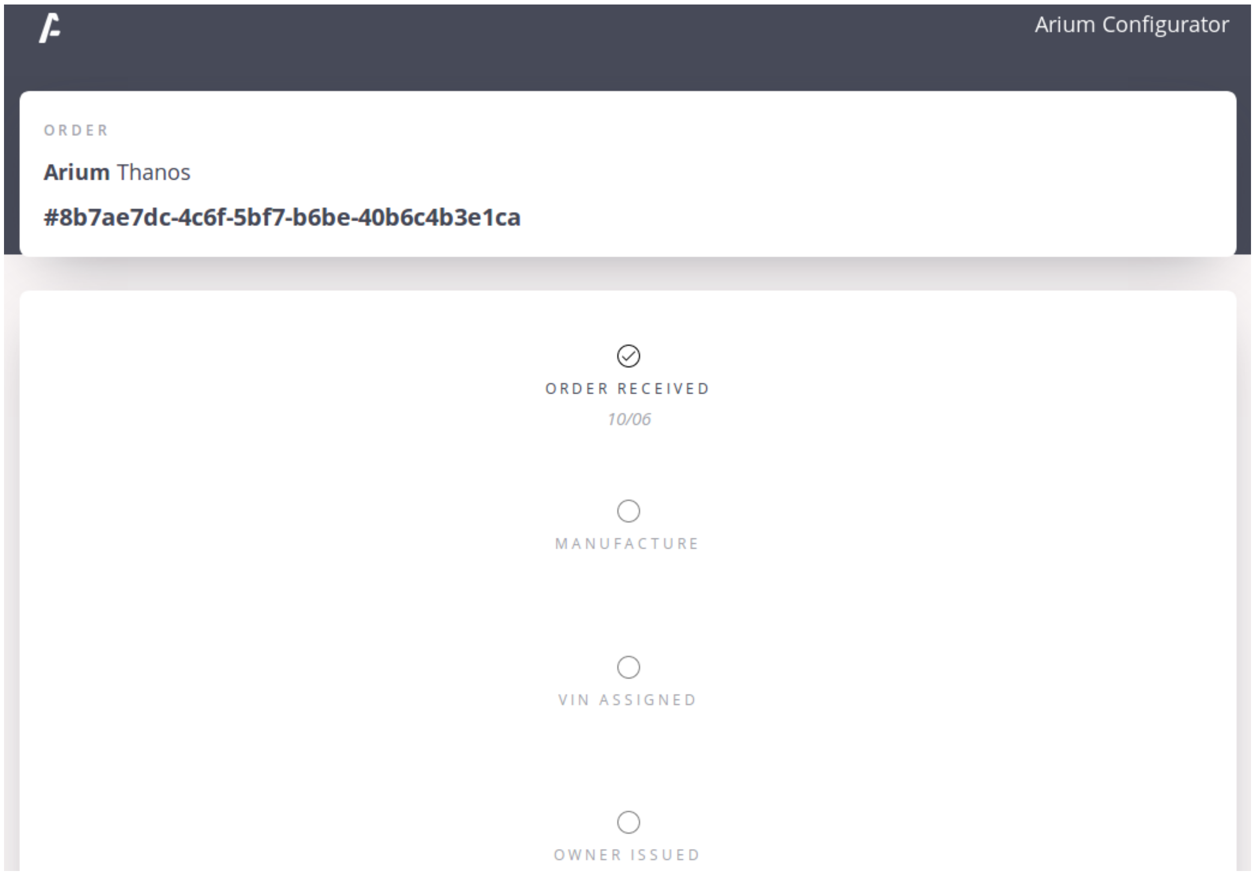
INTERIOR

EXTRAS

__ **7.** Once you have decided on each of Paul's options, click '**Place Order**' at the bottom of the screen:



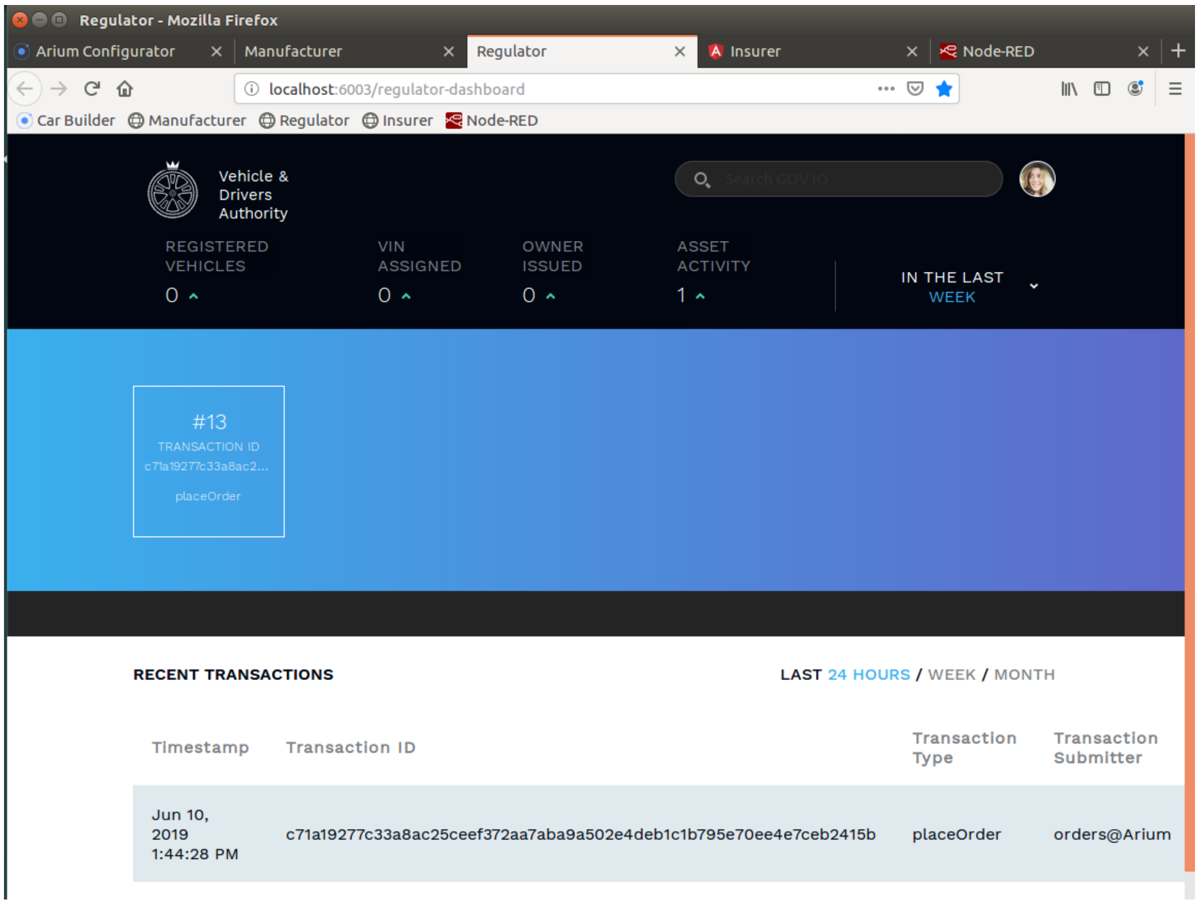
The screen will then change to show that order has been made:



__ **8.** Switch to the “Regulator” tab (localhost:6003/regulator-dashboard).

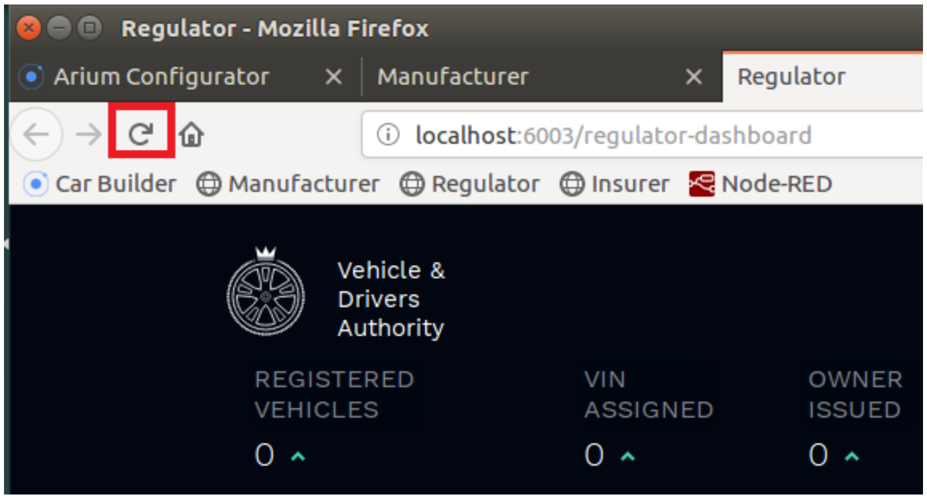
As you will recall, the VDA is the regulator who requires notification of all transactions that occur within the business network. Debbie, who works for the regulator, has a dashboard running on her PC that shows all transactions as they occur.

You will see the VDA dashboard update itself with the latest transaction.

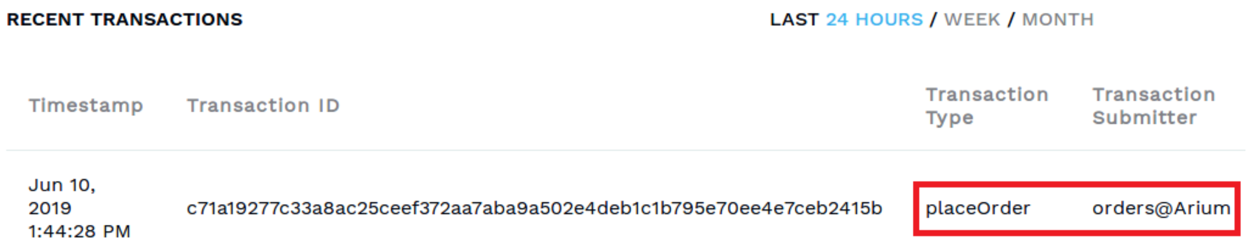


Timestamp	Transaction ID	Transaction Type	Transaction Submitter
Jun 10, 2019 1:44:28 PM	c71a19277c33a8ac25ceef372aa7aba9a502e4deb1c1b795e70ee4e7ceb2415b	placeOrder	orders@Arium

Note: If the screen does not update to show the block, click on the refresh button as shown below:



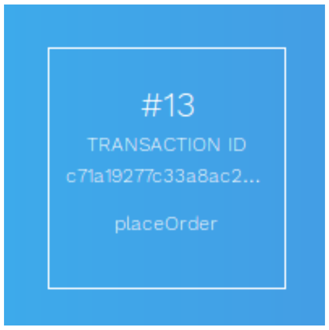
__ **9.** If you look at the "Recent Transactions" log at the bottom of the page, you will see a new transactions listed: a "**placeOrder**" transaction submitted by **orders@Arium**, our manufacturer.



RECENT TRANSACTIONS LAST 24 HOURS / WEEK / MONTH

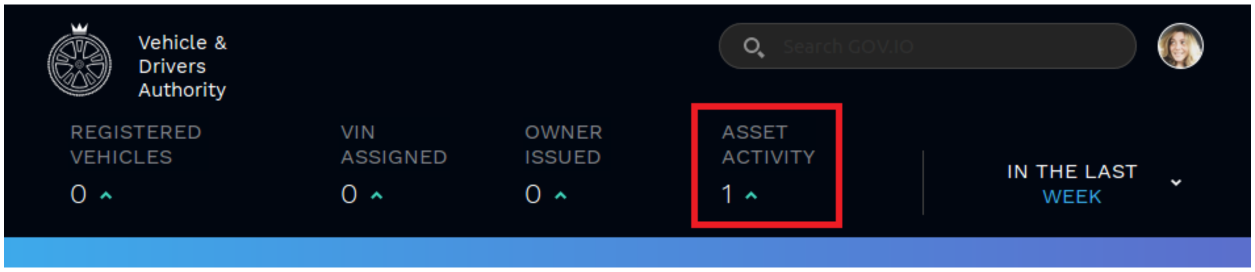
Timestamp	Transaction ID	Transaction Type	Transaction Submitter
Jun 10, 2019 1:44:28 PM	c71a19277c33a8ac25ceef372aa7aba9a502e4deb1c1b795e70ee4e7ceb2415b	placeOrder	orders@Arium

__ **10.** Look in the blue section above this log and you will see this transaction represented graphically as the start of a chain, with the most recent transactions on the right. We will see the number of transactions increase as we work through this lab. This chain is a representation of our blockchain, and the regulator can see everything that is stored on it.

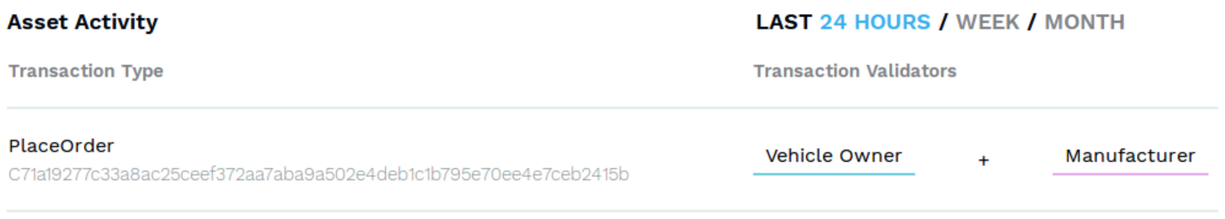


As you will recall, the blockchain is our transaction log which is shared (with appropriate privacy and permissioning) between the participants of our business network. Each block in this chain could potentially actually contain multiple transactions, but in this lab you'll see each unique transaction inside its own block.

__ **11.** Click on 'Asset activity' within the VDA dashboard.



This is an alternative view of the ledger that shows all the transactions that have occurred, and the participants involved.



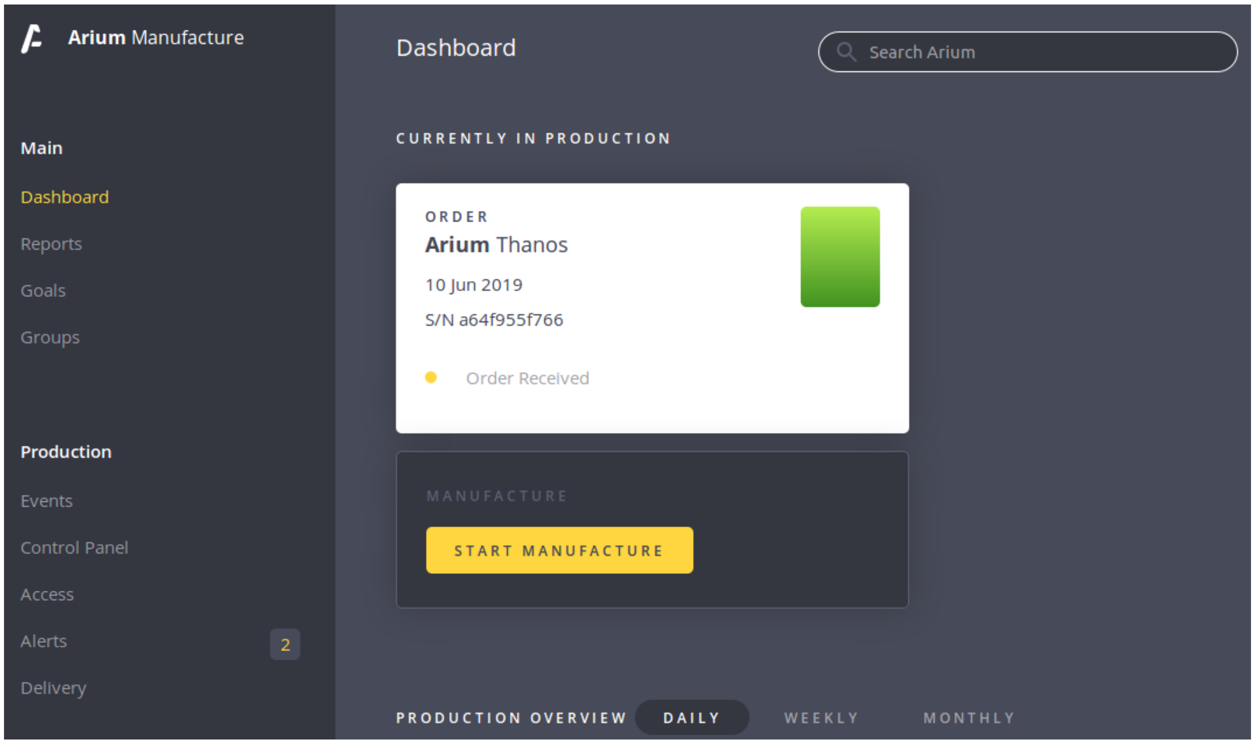
Asset Activity **LAST 24 HOURS / WEEK / MONTH**

Transaction Type	Transaction Validators
PlaceOrder C71a19277c33a8ac25ceef372aa7aba9a502e4deb1c1b795e70ee4e7ceb2415b	Vehicle Owner + Manufacturer

2.2 Manufacturing the car

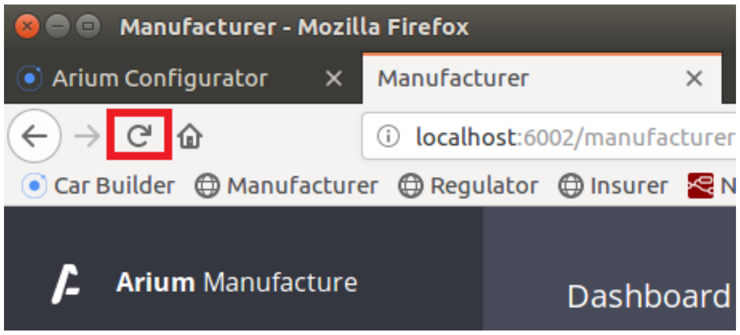
__ **12.** Switch to the "**Manufacturer**" tab:
(http://localhost:6002/manufacturer-dashboard).

This is the dashboard that Mike, who works for Arium, uses. He does not have full visibility into the entire blockchain that the regulator requires but can see the parts of it that pertain to Arium: specifically, he has visibility into all the orders that are coming in so that he can control the manufacturing process.



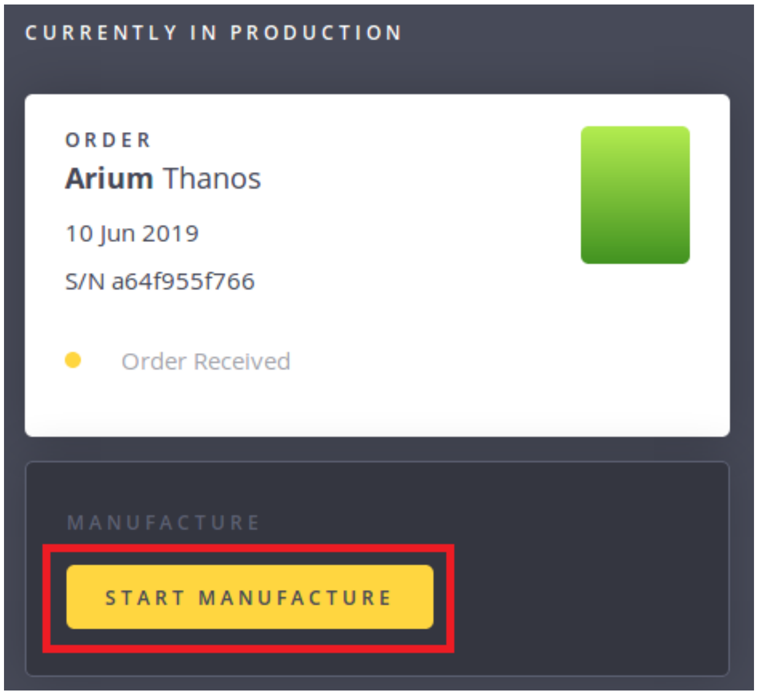
ORDER
Arium Thanos
10 Jun 2019
S/N a64f955f766
Order Received

Note: If the screen does not update to show new order, click on the refresh button as shown below:



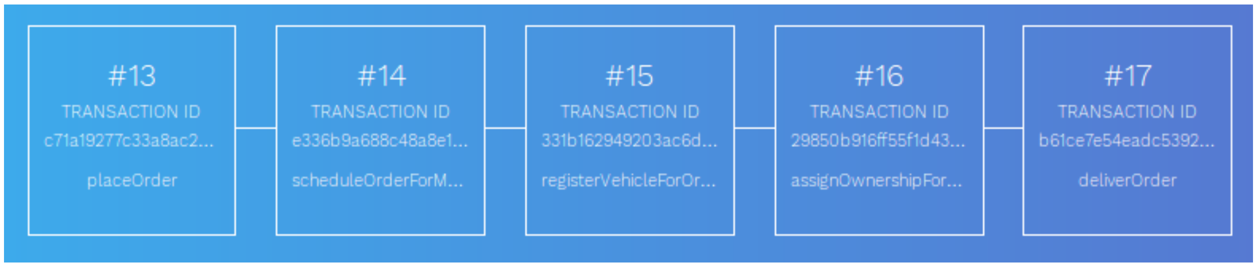
The "**Currently in Production**" section of this page shows those orders that have been received and the cars that have recently been built. The left-most order in this section will be the car that Paul ordered most recently.

__ **13.** Click "**Start Manufacture**" underneath Paul's order to start the business process to build a car. You can immediately see the colour of the car matches the one selected in the order, for example "Alpine Green"



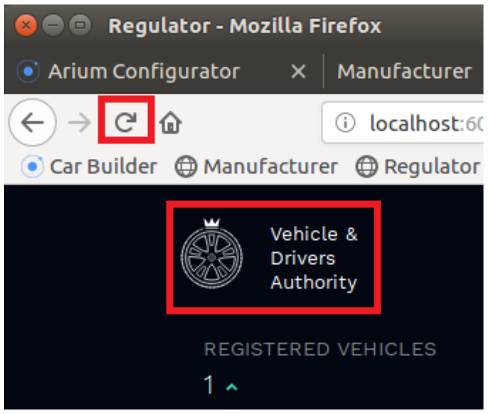
The production process has (of course) been simulated and will take place over the next several seconds; the vehicle will be 'built' and blockchain transactions submitted that record status at key milestones of the production process. In a real network, different automated plant systems will trigger these events, which are signed off by the manufacturer, and the issuance of the Vehicle Identification Number might be signed off by the regulator.

__ **14.** As the car is being built, switch back to the VDA **Regulator** dashboard tab to see these key milestones being represented on the unfiltered blockchain.

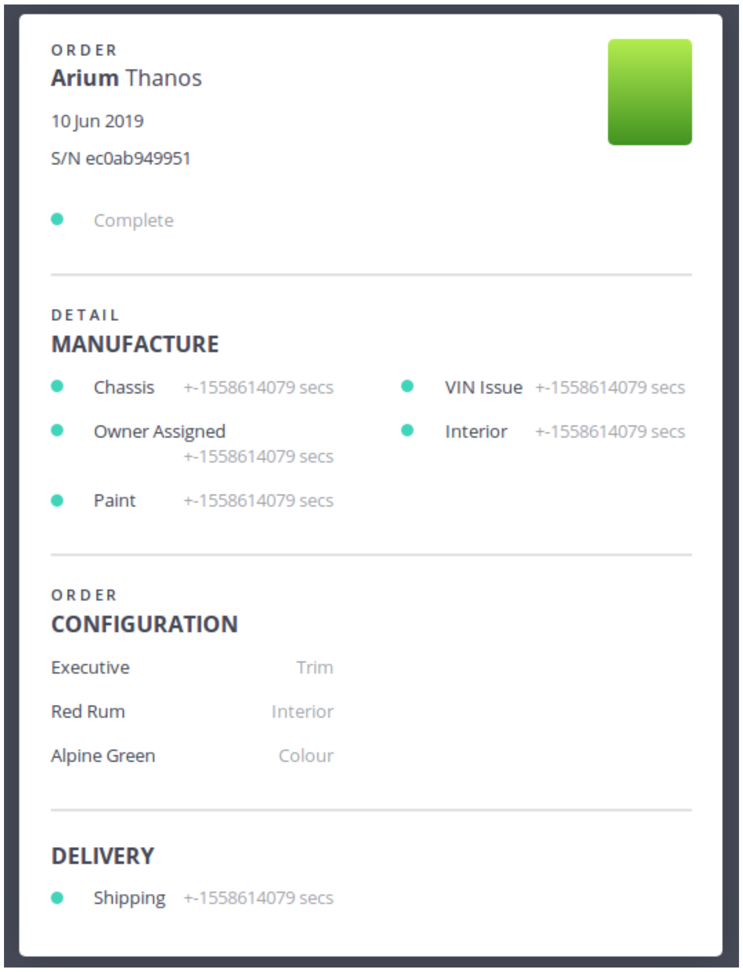


#13	#14	#15	#16	#17
TRANSACTION ID	TRANSACTION ID	TRANSACTION ID	TRANSACTION ID	TRANSACTION ID
c71a19277c33a8ac2...	e336b9a688c48a8e1...	331b162949203ac6d...	29850b916ff55f1d43...	b61ce7e54eadc5392...
placeOrder	scheduleOrderForM...	registerVehicleForOr...	assignOwnershipFor...	deliverOrder

Note: You will need to click on the VDA logo to see the main screen again, and you may need to refresh the screen again to see the new blocks as shown below:



__ **15.** Also note how the Manufacturer's view changes as the vehicle is being built, with icons changing to green as those parts of the process are completed.



ORDER
Arium Thanos
10 Jun 2019
S/N ec0ab949951

● Complete

DETAIL
MANUFACTURE

● Chassis	+-1558614079 secs	● VIN Issue	+-1558614079 secs
● Owner Assigned	+-1558614079 secs	● Interior	+-1558614079 secs
● Paint	+-1558614079 secs		

ORDER
CONFIGURATION

Executive	Trim
Red Rum	Interior
Alpine Green	Colour

DELIVERY

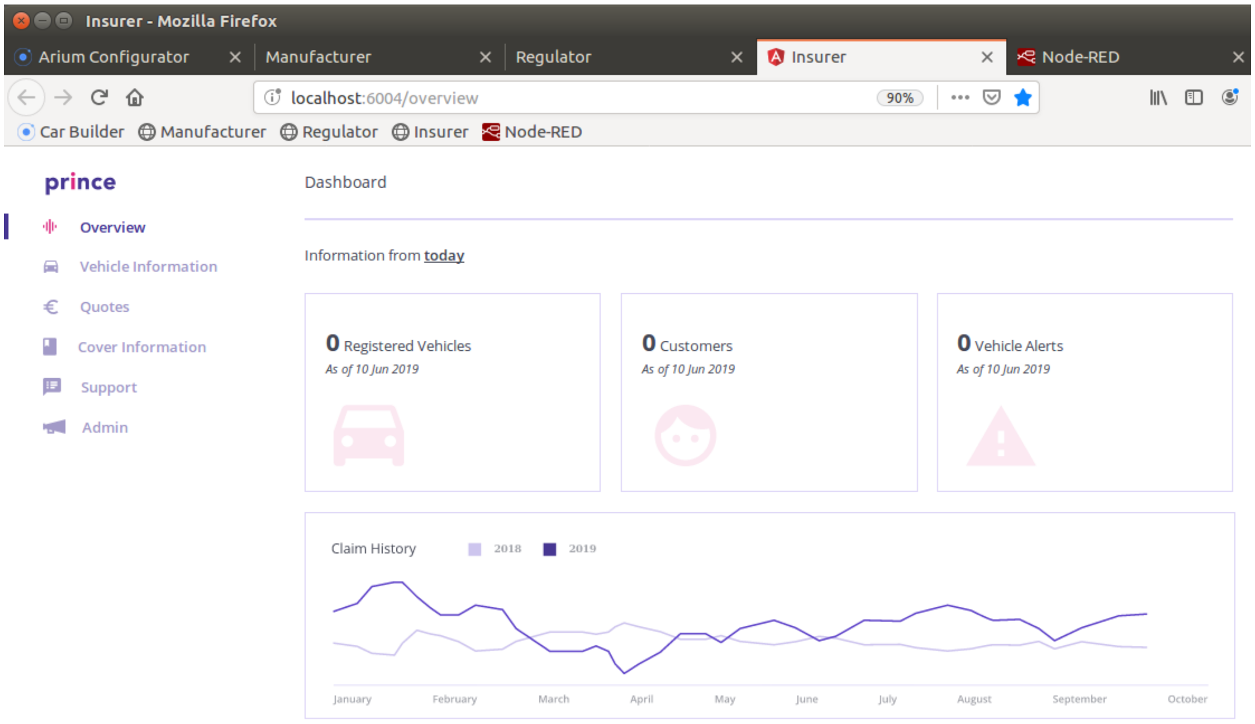
● Shipping +-1558614079 secs

2.3 Insuring the car

As Paul takes ownership of his new car, we will give him the option to insure it directly in his app. His insurance company offers a discount if he chooses to provide the insurance company with frequent details of the car's location and other things using sensors located in the car.

The manufacturer fits the car with a collection of IoT devices, including GPS, air and engine temperature sensors, acceleration information and light information, which can give the insurer information on how the car is being driven, and potentially alert relevant parties if the car is involved in a crash.

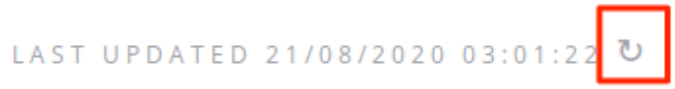
__ **16.** Switch to the **Insurer** dashboard tab (localhost:6004/overview). Ensure that the 'Overview' tab is selected.



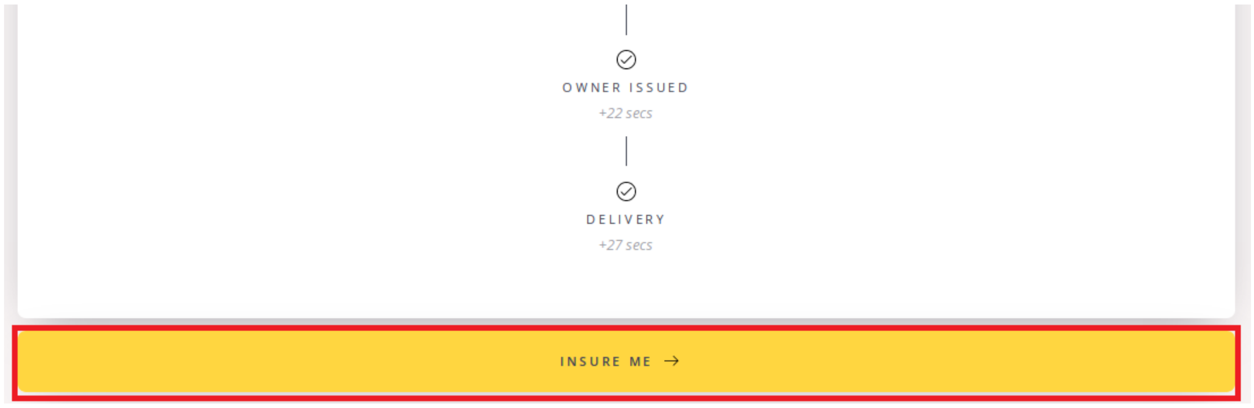
Tommen works for Prince Insurance and this is his dashboard. He requires another subset of information from the blockchain and this view is represented here. He can see information on the cars for which his company is an insurer and can also approve new polices. (In reality, this latter part can be automated.)

Currently as you can see no cars are insured, so let's change that.

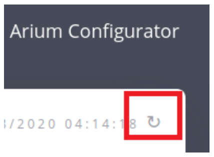
__ **17.** Switch back to the app Paul uses – the “**Arium Configurator**” tab. IMPORTANT – If the screen needs to be refreshed because the status has not been updated, use the refresh arrow next to the LAST UPDATED text in the upper righthand corner.



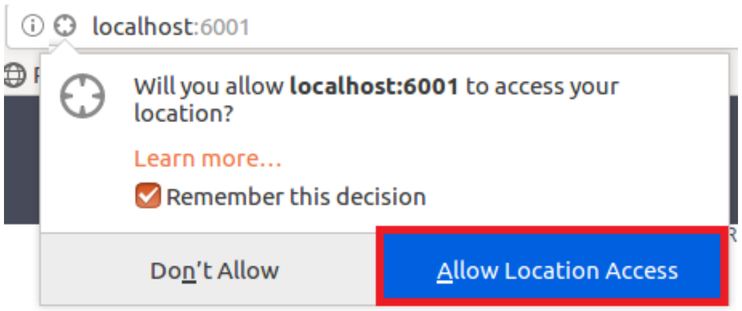
After the car has been delivered by the manufacturer scroll down to the bottom and click the **“Insure Me”** button:



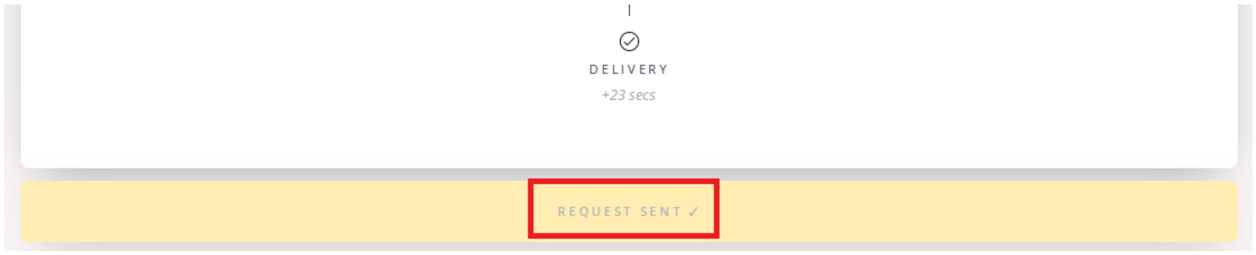
Note: If the screen does not update to show the “**Insure Me**” button, click on the refresh button on the top-right of this page:



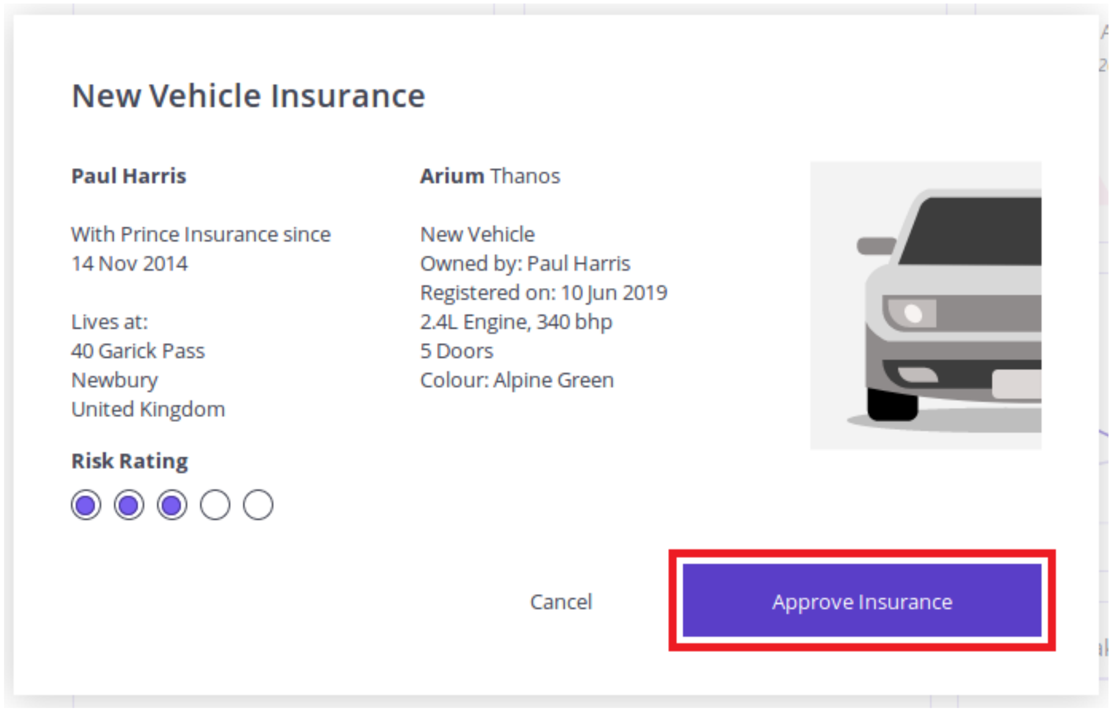
__ **18.** Click “**Allow Location Access”** if a popup appears; Paul is willing to share the IoT device’s location with the insurance company.



The “**Insure Me**” button will then change to show that the request has been sent:



__ **19.** Switch back to the “**Insurer**” tab view and click the “**Approve Insurance**” button:



New Vehicle Insurance

Paul Harris

With Prince Insurance since
14 Nov 2014

Lives at:
40 Garick Pass
Newbury
United Kingdom

Arium Thanos

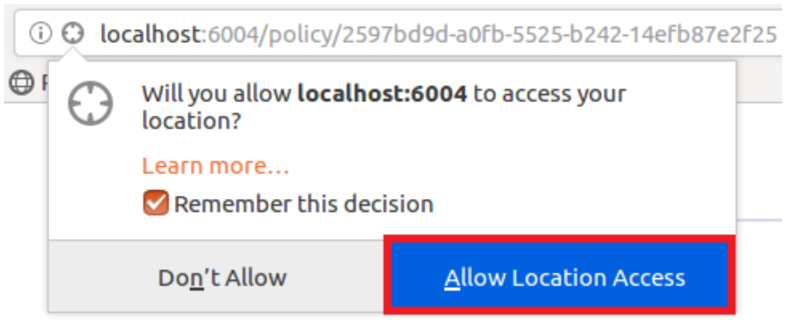
New Vehicle
Owned by: Paul Harris
Registered on: 10 Jun 2019
2.4L Engine, 340 bhp
5 Doors
Colour: Alpine Green

Risk Rating

● ● ● ○ ○

Note: If the screen does not update to show the “**New Vehicle Insurance**” panel, click on the refresh button on the browser tab.

__ **20.** Click “**Allow Location Access”** if a popup appears.



localhost:6004/policy/2597bd9d-a0fb-5525-b242-14efb87e2f25

Will you allow **localhost:6004** to access your location?

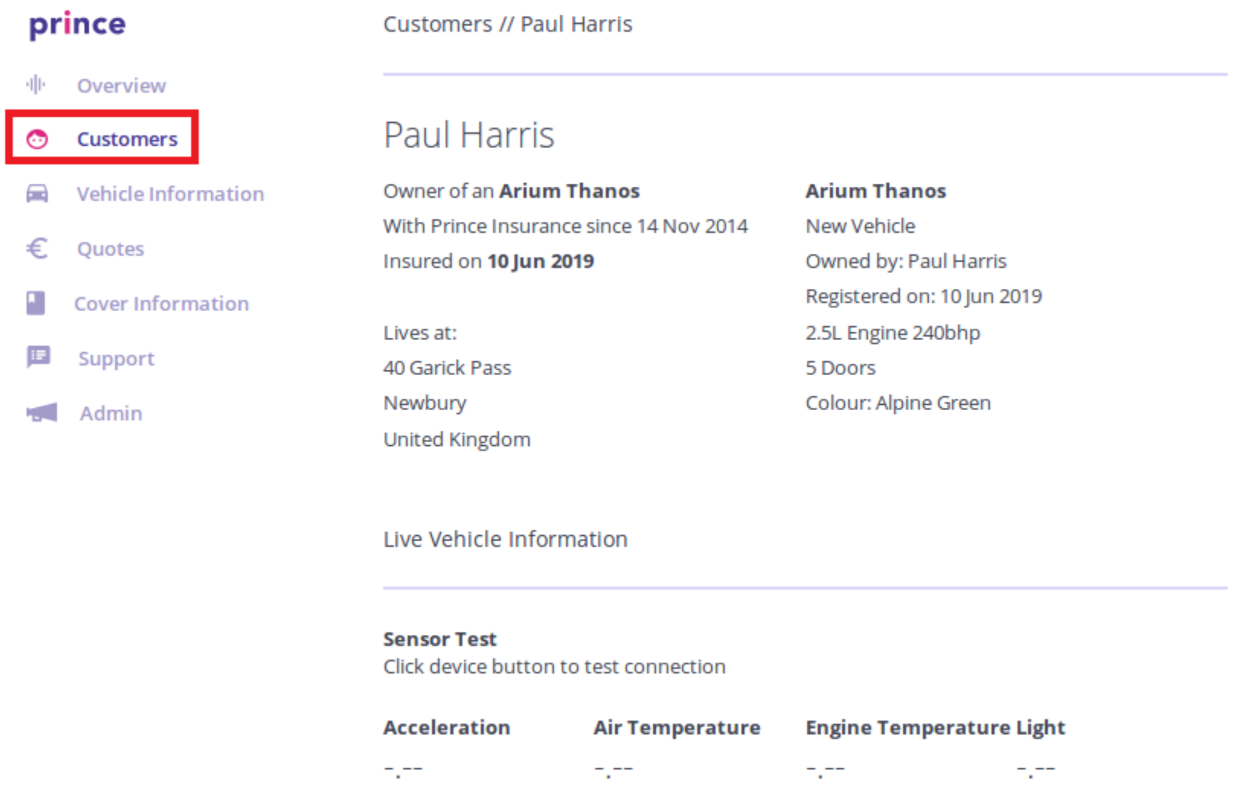
Learn more…

☑ Remember this decision

Once the approval is logged on the blockchain, Paul is now insured by the Prince insurance company.

You can switch back to the **Regulator** tab to see the **createPolicy** transaction if you wish

__ **21.** Review the '**Customers'** tab on the **Insurers** screen to see details of Paul's policy.



Customers // Paul Harris

Paul Harris

Owner of an **Arium Thanos**
With Prince Insurance since 14 Nov 2014
Insured on **10 Jun 2019**

Lives at:
40 Garick Pass
Newbury
United Kingdom

Arium Thanos
New Vehicle
Owned by: Paul Harris
Registered on: 10 Jun 2019
2.5L Engine 240bhp
5 Doors
Colour: Alpine Green

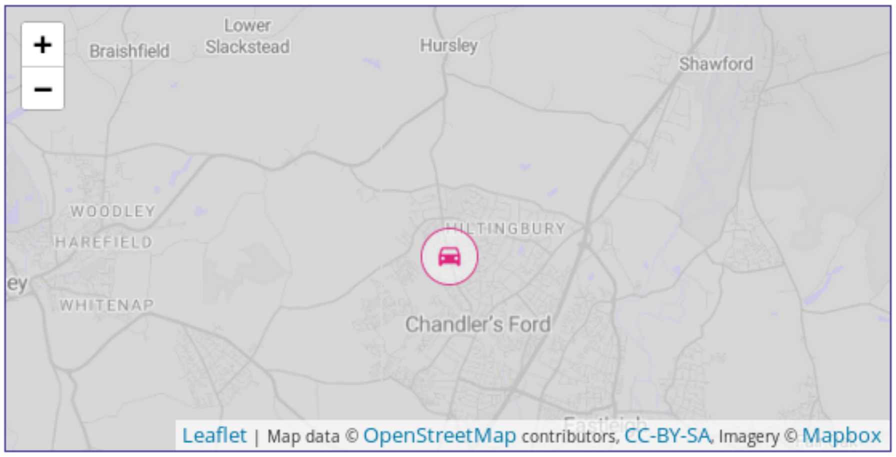
Live Vehicle Information

Sensor Test
Click device button to test connection

Acceleration	**Air Temperature**	**Engine Temperature Light**	
-.--	-.--	-.--	-.--

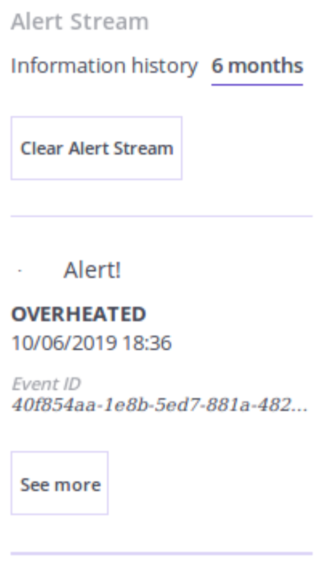
At the top of the page you can see basic details of Paul's policy including his address and car information. Underneath this is the set of raw readings from the IoT devices attached to Paul's car. This is useful information for debugging; in reality the blockchain is not used to share complete data streams from the IoT sensors as the amount of data is too great and is not relevant to be shared in its entirety.

Location



However, what would be relevant is the analysis of key events in the IoT stream. For example, if the acceleration is shown to be greater than (for example) 2G, this might indicate a crash event that the insurer might care about.

This is shown as a set of alerts on the right-hand side of the **Insurer's** customer view:


Alert Stream

Information history 6 months

Alert!

OVERHEATED

10/06/2019 18:36

Event ID
40f854aa-1e8b-5ed7-881a-482...

Without a real sensor tag connected into the application, the information displayed here is empty. In the next section we will inject data into the application using internet of things integration.

It is possible to connect a real sensor tag so that its information is displayed in the **Insurer** view; we have tested using a Texas Instruments SimpleLink Bluetooth SensorTag. To use this, you need to download the TI SimpleLink Starter app to a nearby mobile device, use it to discover the sensor via Bluetooth, note down the unique address of the tag and enable the "Push to cloud" option to submit the sensor readings so that they can be read by the IBM Watson IoT platform. Then you need to update the "IBM IoTP Test Device" node in the Node-RED flow to monitor the readings from the unique address of the tag from the cloud. Remember to redeploy the Node.RED flow.

3 Cool Stuff

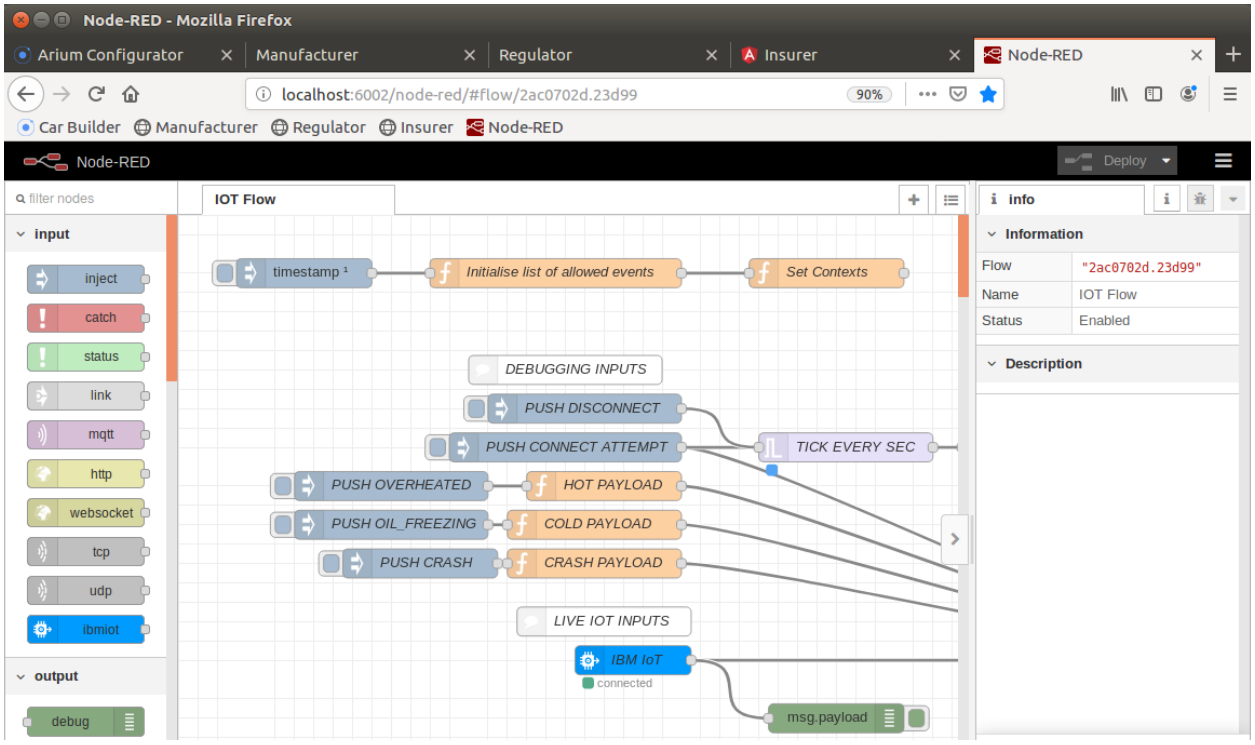
In this section, we're going to look at how the scenario can be enhanced by bringing blockchain together with internet of things and analytics.

3.1 Internet of Things Integration

We will start by looking at how sensor data from the car makes its way into the blockchain. To do this we will use an integration tool called Node-RED. This includes a graphical interface to describe how data flows from input sources (e.g. a sensor) to output sources (e.g. the blockchain).

Node-RED has connectors for sending data to, and receiving data from, a blockchain. It also has connectors for receiving data from the IBM Watson IoT platform (for sensor tag integration). We can also generate fake sensor data for testing, in the absence of a physical sensor device.

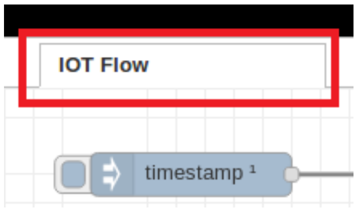
__ **22.** Switch to the **Node-RED** tab (http://localhost:6002/node-red/).



The main window shows the flow of how data from devices is mapped to blockchain events. The tabs along the top show the different flows that are deployed. Down the left hand side, you can see the available connectors for wiring into the flow. The right hand-side contains the properties of the selected connector and debug information.

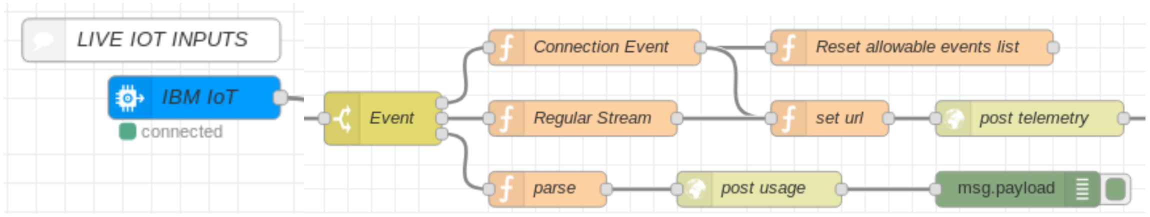
Note: If you make any changes to the flow, you need to press the "Deploy" button in the top right to let them take effect.

__ **23.** Ensure that the "IoT Flow" tab in Node-RED is selected:

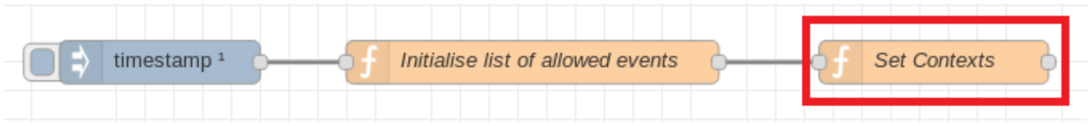


There are some interesting things to note in this flow.

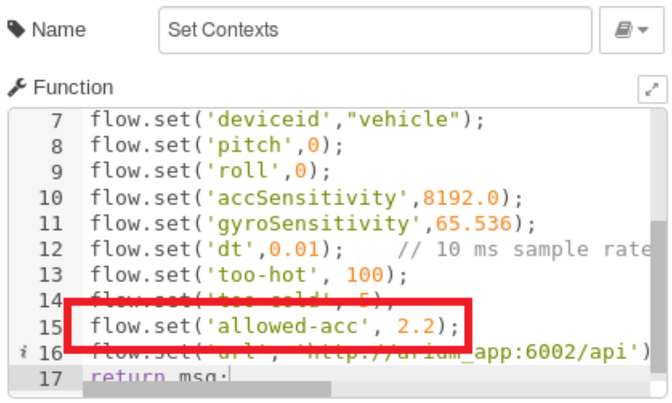
__ **24.** Look at the "**IBM IoT**" Input node under "**Live IOT Inputs**" section. This takes readings from a real sensor device and publishes any interesting events to the blockchain through the other nodes wired downstream.



__ **25.** Double click the "**Set Contexts**" connector near the top right of the flow:



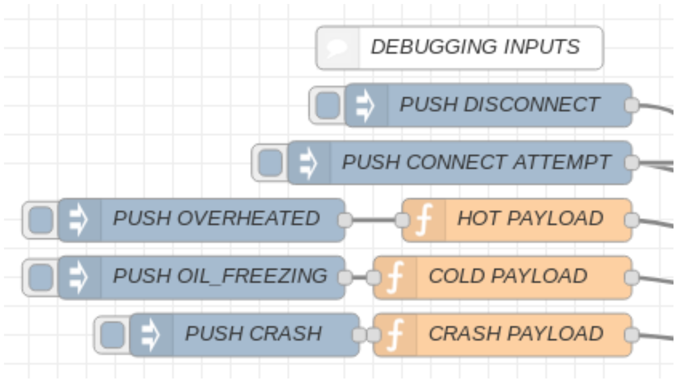
This shows the thresholds for sending interesting events to the blockchain. For example, if the acceleration is greater than 2.2G, this causes a crash event to be sent to the blockchain.



```
flow.set('deviceid',"vehicle");
flow.set('pitch',0);
flow.set('roll',0);
flow.set('accSensitivity',8192.0);
flow.set('gyroSensitivity',65.536);
flow.set('dt',0.01);    // 10 ms sample rate
flow.set('too-hot', 100);
flow.set('too-cold', 5);
flow.set('allowed-acc', 2.2);
flow.set('url', 'http://arium_app:6002/api')
return msg;
```

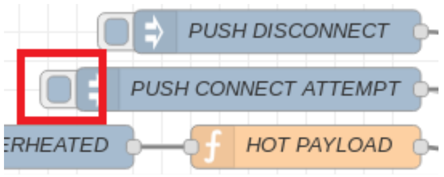
__ **26.** Click “**Cancel**” to switch back to the IOT flow.

__ **27.** Look at the set of connectors next to the “DEBUGGING INPUTS” section: PUSH DISCONNECT, PUSH CONNECT ATTEMPT, PUSH OVERHEATED, PUSH OIL_FREEZING and PUSH CRASH:

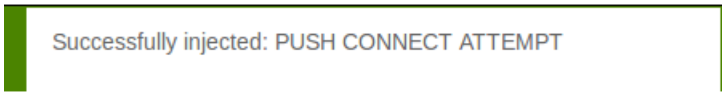


These connectors allow us to simulate an interesting event occurring, in the absence of a real device.

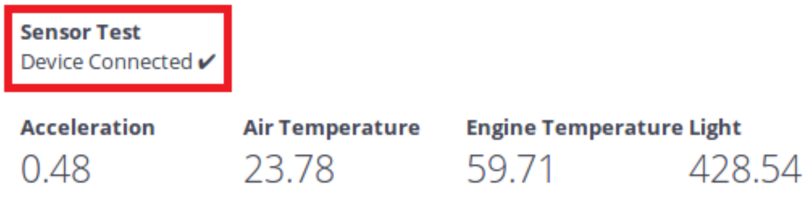
__ **28.** Click the rounded square button next to the PUSH CONNECT ATTEMPT connector:



You should see a message saying that data was successfully injected into the flow:

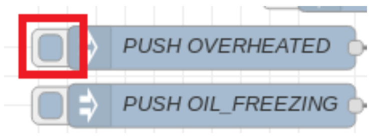

Successfully injected: PUSH CONNECT ATTEMPT

__ **29.** Switch to the **Insurer** tab, and notice under “**Sensor Test**” that the vehicle sensor is now connected and test data is being injected into the flow:

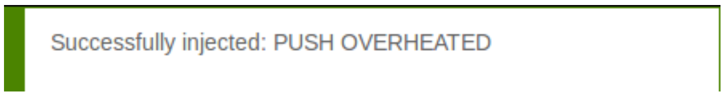

Sensor Test
Device Connected ✔

Acceleration	Air Temperature	Engine Temperature	Light
0.48	23.78	59.71	428.54

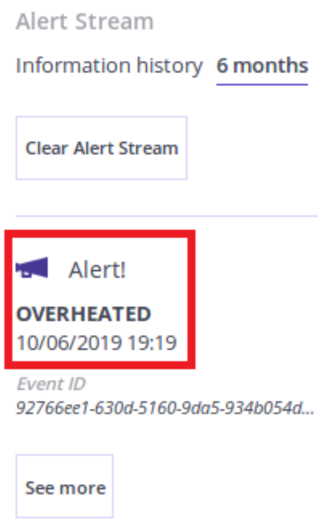
__ **30.** Switch back to the **Node-RED** tab, and click the button next to the PUSH OVERHEATED node to send an event to the blockchain which denotes Paul’s engine overheating:



You should again see a “Successfully injected” message:


Successfully injected: PUSH OVERHEATED

__ **31.** In the Insurer view you should see an alert that reveals this event to the insurer:



Alert!

OVERHEATED

10/06/2019 19:19

Event ID

92766ee1-630d-5160-9da5-934b054d...

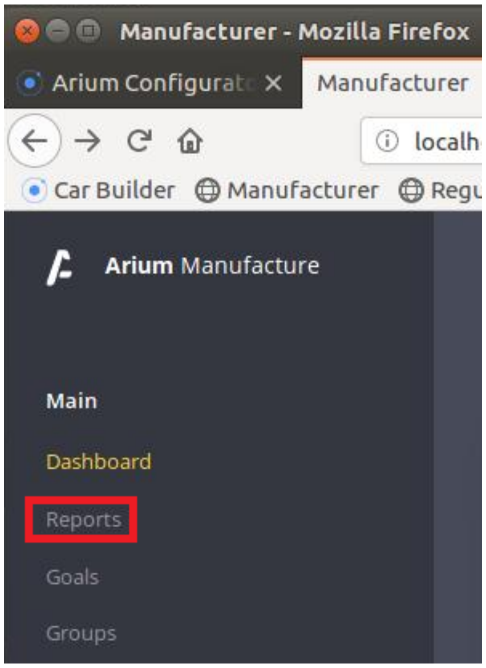
__ **32.** Try invoking the other events too (OIL_FREEZING, CRASH) to see their effect as well.

More details on the IBM Watson IoT Platform can be found at this address: https://internetofthings.ibmcloud.com/

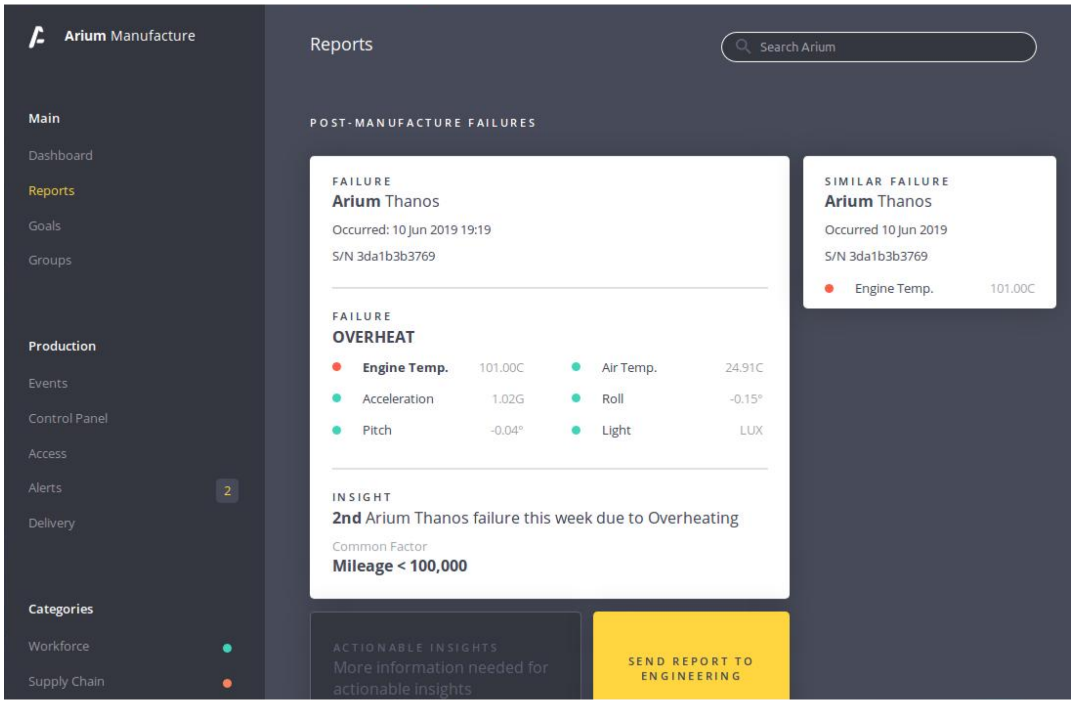
3.2 Analytics

It is possible to use the information stored on the blockchain to provide insight on aggregate usage patterns to interested authorized parties. This gives the power of data analytics on top of the benefits of a blockchain, as a verifiably clean source of information to analyse.

__ **33.** Switch back to the **Manufacturer** view tab and click the "**Reports**" link:



The engine overheated events show in this view. These events were captured in the blockchain and the manufacturer role has permission see this type of event. The manufacturer wishes to detect trends in engine overheated failures in order to determine if a factory defect is causing this condition:



POST-MANUFACTURE FAILURES

FAILURE
Arium Thanos
Occurred: 10 Jun 2019 19:19
S/N 3da1b3b3769

FAILURE
OVERHEAT

●	Engine Temp.	101.00C	●	Air Temp.	24.91C
●	Acceleration	1.02G	●	Roll	-0.15°
●	Pitch	-0.04°	●	Light	LUX

INSIGHT
2nd Arium Thanos failure this week due to Overheating
Common Factor
Mileage < 100,000

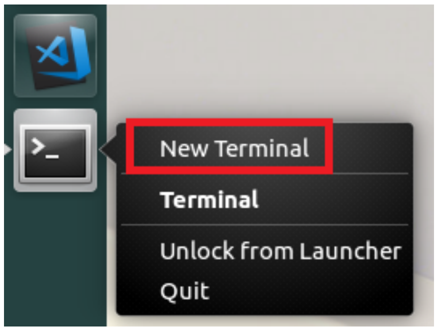
ACTIONABLE INSIGHTS
More information needed for actionable insights

SIMILAR FAILURE
Arium Thanos
Occurred 10 Jun 2019
S/N 3da1b3b3769
● Engine Temp. 101.00C

3.3 Under the Hood (*Optional*)

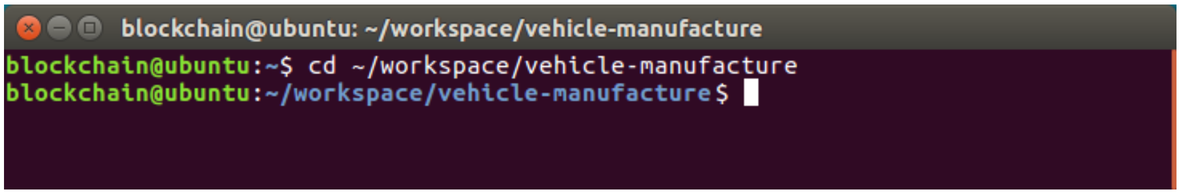
The participants in this lab are connected together using a real Hyperledger Fabric network. If you are a software developer, try running these additional steps to explore one of the smart contracts that is running behind the scenes.

__ **34.** From the desktop, right click on the **Terminal** icon and select "**New Terminal**" to open a new terminal:



__ **35.** Enter this command (or cut and paste it):

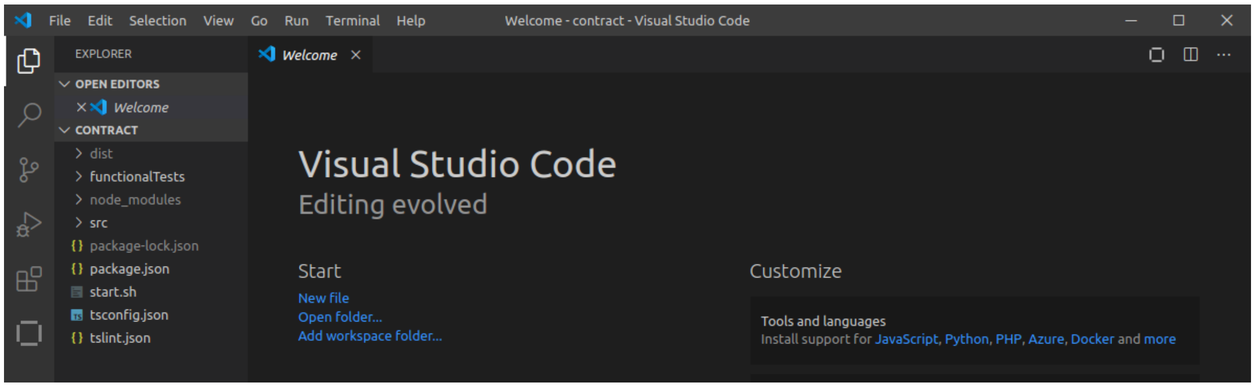
```
cd ~/workspace/vehicle-manufacture
```



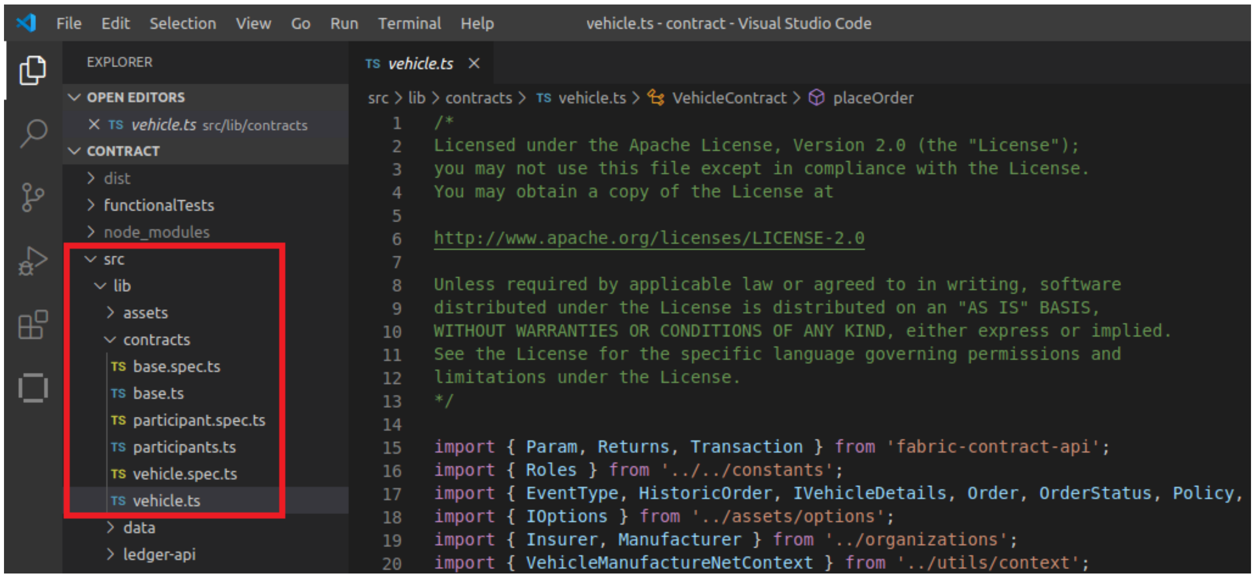
```
blockchain@ubuntu: ~/workspace/vehicle-manufacture
blockchain@ubuntu:~$ cd ~/workspace/vehicle-manufacture
blockchain@ubuntu:~/workspace/vehicle-manufacture$
```

__ **36.** Enter this command (or cut and paste it) to open the folder containing the smart contract source code in the VS Code editor:

```
code contract
```



__ **37.** Expand the **src/lib/contracts** folder and double click on **vehicle.ts** to open it



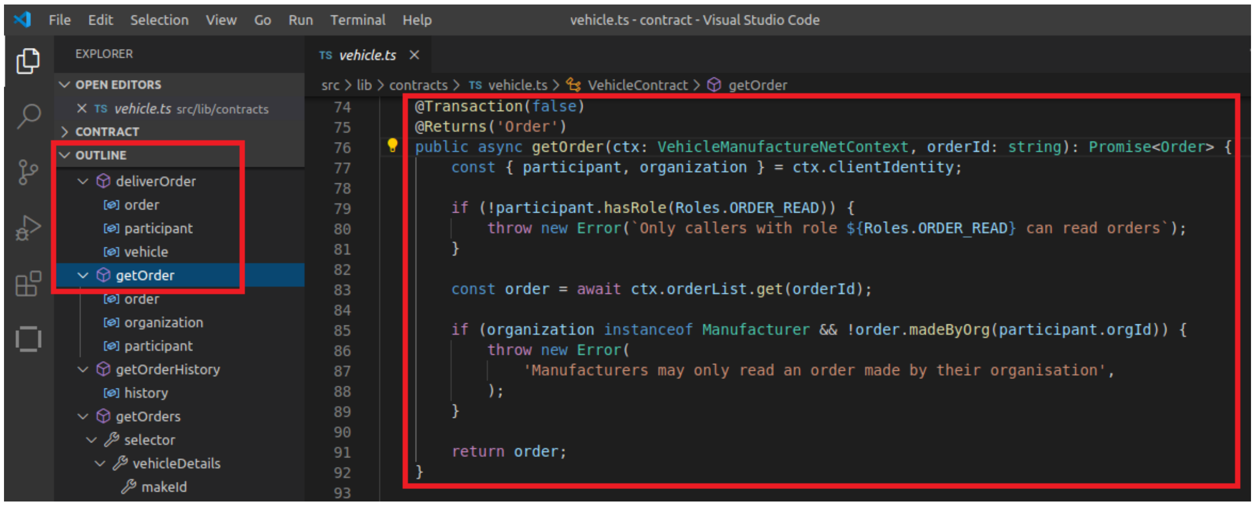
```
/*
Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
*/

import { Param, Returns, Transaction } from 'fabric-contract-api';
import { Roles } from '../../constants';
import { EventType, HistoricOrder, IVehicleDetails, Order, OrderStatus, Policy,
import { IOptions } from '../assets/options';
import { Insurer, Manufacturer } from '../organizations';
import { VehicleManufactureNetContext } from '../utils/context';
```

__ **38.** In the "**Outline**" view scroll down to click on the "**getOrder**" transaction and look at how it is implemented. We can see how the **orderId** which is passed in is used to find an order to return to the caller.



```
@Transaction(false)
@Returns('Order')
public async getOrder(ctx: VehicleManufactureNetContext, orderId: string): Promise<Order> {
    const { participant, organization } = ctx.clientIdentity;

    if (!participant.hasRole(Roles.ORDER_READ)) {
        throw new Error(`Only callers with role ${Roles.ORDER_READ} can read orders`);
    }

    const order = await ctx.orderList.get(orderId);

    if (organization instanceof Manufacturer && !order.madeByOrg(participant.orgId)) {
        throw new Error(
            'Manufacturers may only read an order made by their organisation',
        );
    }

    return order;
}
```

Feel free to look at the other transactions in this file whilst you still have time remaining.

__ **39.** From the vs

4 Next Steps

In this lab you have experienced a live blockchain solution through the eyes of four participants of a vehicle network: a **buyer/owner**, **manufacturer**, **regulator** and **insurer**. A blockchain can be used to great effect in this business network because there is a clear need to share information and value in participants being able to trust the information they see.

Where you go from here is up to you.

If you have a technical background, consider finding out more about Hyperledger Fabric and the IBM Blockchain Platform development experience to implement your first use-case. You can also sign up to the IBM Blockchain Platform to play more with the blockchain technology here: “https://www.ibm.com/blockchain/platform”

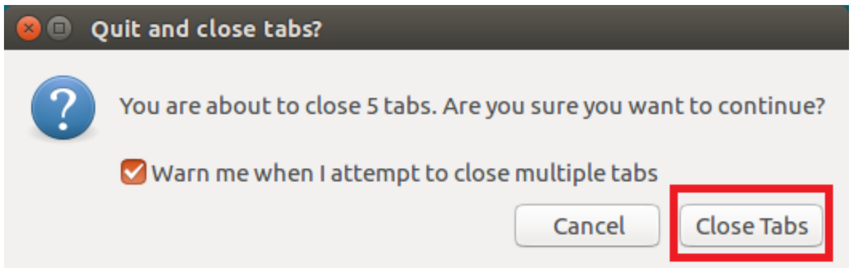
If you are interested in the potential benefits of blockchain in your business, IBM has a bunch of services that can help. Start by going to “www.ibm.com/blockchain”.

Congratulations on completing the lab!

Appendix A. Restarting the sample application

The virtual machine used in the lab is set to start the application, blockchain and associated services automatically. If for any reason, they have broken or need restarting, open a terminal and type:

1) Close the Firefox browser window and all tabs:

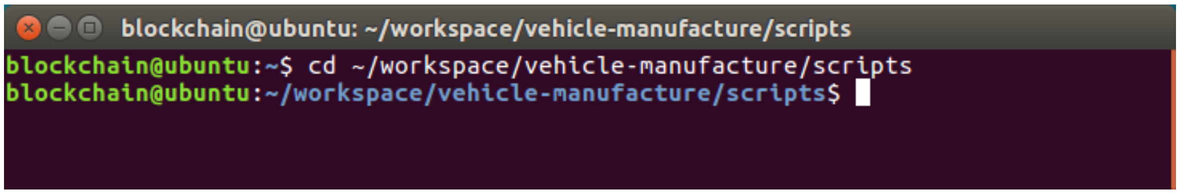


Quit and close tabs?

You are about to close 5 tabs. Are you sure you want to continue?

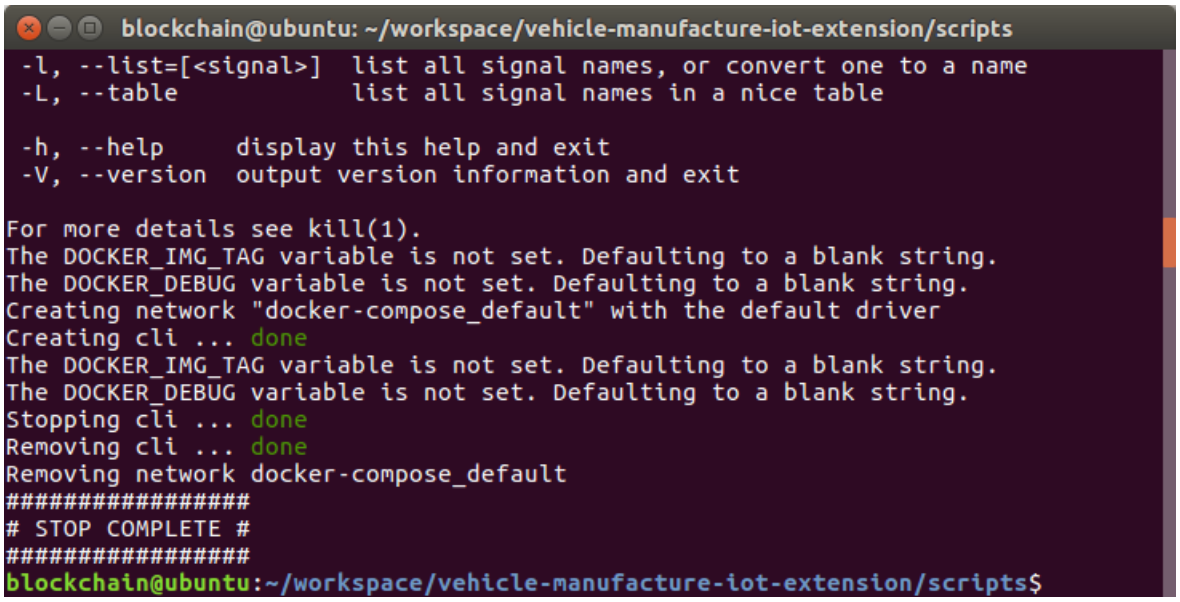
Warn me when I attempt to close multiple tabs

2) Open a new terminal and run:
`cd ~/workspace/vehicle-manufacture/scripts`



```
blockchain@ubuntu: ~/workspace/vehicle-manufacture/scripts
blockchain@ubuntu:~$ cd ~/workspace/vehicle-manufacture/scripts
blockchain@ubuntu:~/workspace/vehicle-manufacture/scripts$
```

3) Enter the command:
./stop.sh

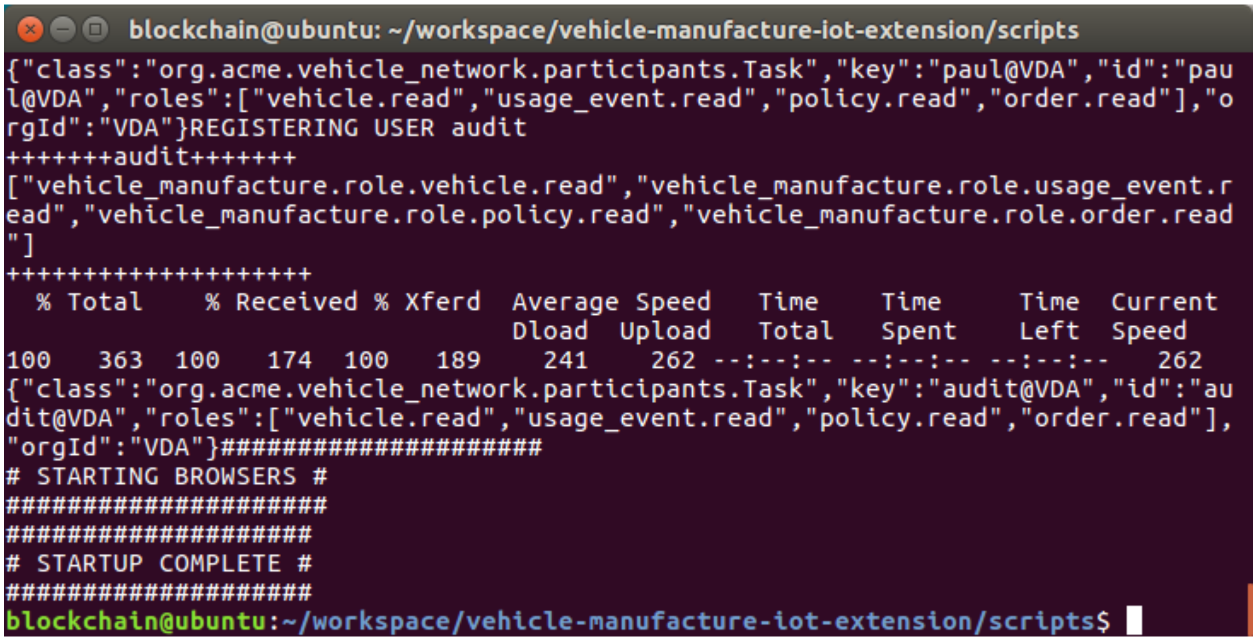


```
blockchain@ubuntu: ~/workspace/vehicle-manufacture-iot-extension/scripts
 -l, --list=[<signal>]  list all signal names, or convert one to a name
 -L, --table            list all signal names in a nice table

 -h, --help     display this help and exit
 -V, --version  output version information and exit

For more details see kill(1).
The DOCKER_IMG_TAG variable is not set. Defaulting to a blank string.
The DOCKER_DEBUG variable is not set. Defaulting to a blank string.
Creating network "docker-compose_default" with the default driver
Creating cli ... done
The DOCKER_IMG_TAG variable is not set. Defaulting to a blank string.
The DOCKER_DEBUG variable is not set. Defaulting to a blank string.
Stopping cli ... done
Removing cli ... done
Removing network docker-compose_default
#################
# STOP COMPLETE #
#################
blockchain@ubuntu:~/workspace/vehicle-manufacture-iot-extension/scripts$
```

4) Enter the command:
./start.sh



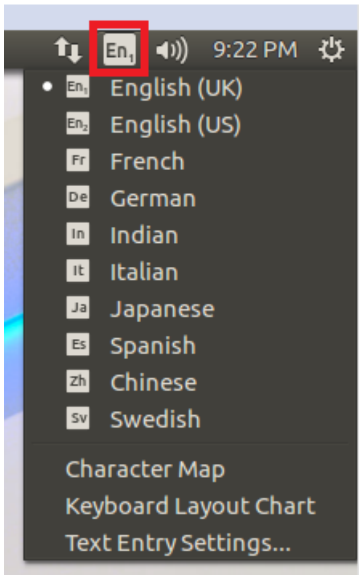
```
blockchain@ubuntu: ~/workspace/vehicle-manufacture-iot-extension/scripts
{"class":"org.acme.vehicle_network.participants.Task","key":"paul@VDA","id":"pau
l@VDA","roles":["vehicle.read","usage_event.read","policy.read","order.read"],"o
rgId":"VDA"}REGISTERING USER audit
+++++++audit+++++++
["vehicle_manufacture.role.vehicle.read","vehicle_manufacture.role.usage_event.r
ead","vehicle_manufacture.role.policy.read","vehicle_manufacture.role.order.read
"]
+++++++++++++++++++++
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   363  100   174  100   189    241    262 --:--:-- --:--:-- --:--:--   262
{"class":"org.acme.vehicle_network.participants.Task","key":"audit@VDA","id":"au
dit@VDA","roles":["vehicle.read","usage_event.read","policy.read","order.read"],
"orgId":"VDA"}#####################
# STARTING BROWSERS #
#####################
####################
# STARTUP COMPLETE #
####################
blockchain@ubuntu:~/workspace/vehicle-manufacture-iot-extension/scripts$
```

The application should take around five minutes to restart and Firefox will reopen showing the tabs.

Appendix B. Changing the keyboard language

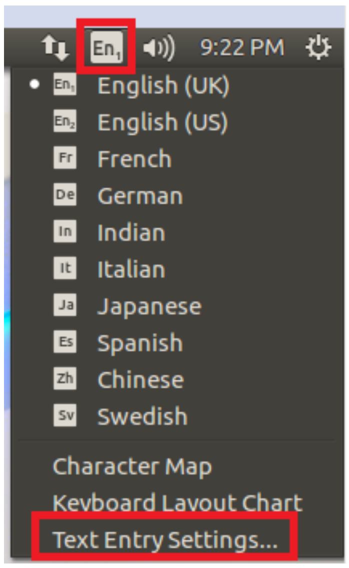
To change the keyboard language to enable you to use laptops with keyboards from different countries follow these steps:

__ **1.** Click on the En icon in the top right and look at the list of languages shown:

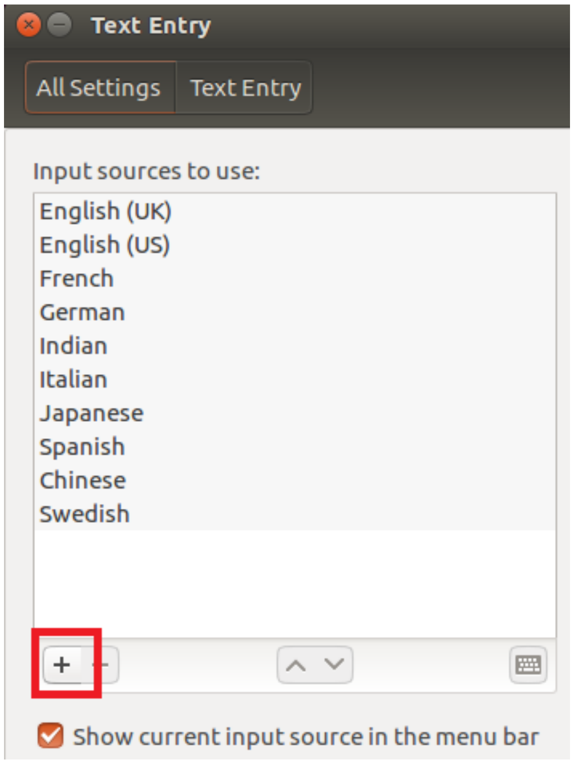


If your required language is present, simply select your language of choice and you are done; skip the following steps.

__ **2.** If your required language is not present, select **Text Entry Settings...** from the list:



__ **3.** **Select** the + symbol to add a new language:

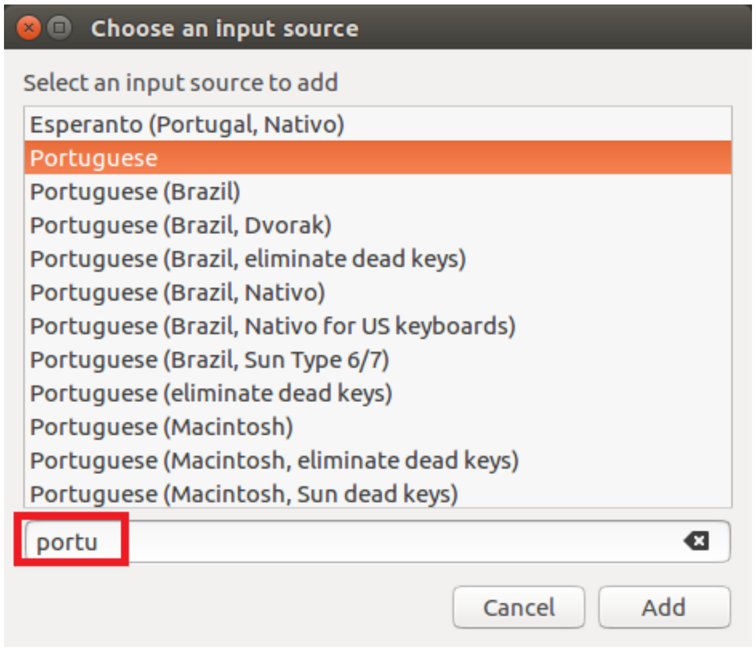


Input sources to use:

English (UK)
English (US)
French
German
Indian
Italian
Japanese
Spanish
Chinese
Swedish

Show current input source in the menu bar

__ **4.** Start to type your language until you see your language's name appear – we will add Portuguese as an example here:

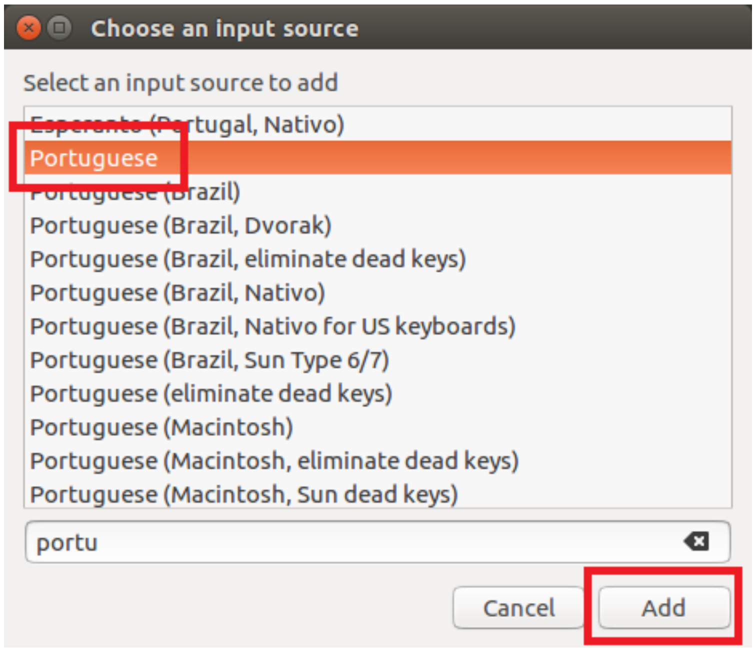


Select an input source to add

Esperanto (Portugal, Nativo)
Portuguese
Portuguese (Brazil)
Portuguese (Brazil, Dvorak)
Portuguese (Brazil, eliminate dead keys)
Portuguese (Brazil, Nativo)
Portuguese (Brazil, Nativo for US keyboards)
Portuguese (Brazil, Sun Type 6/7)
Portuguese (eliminate dead keys)
Portuguese (Macintosh)
Portuguese (Macintosh, eliminate dead keys)
Portuguese (Macintosh, Sun dead keys)

portu

__ **5.** Select your chosen keyboard and click ‘**Add**’:

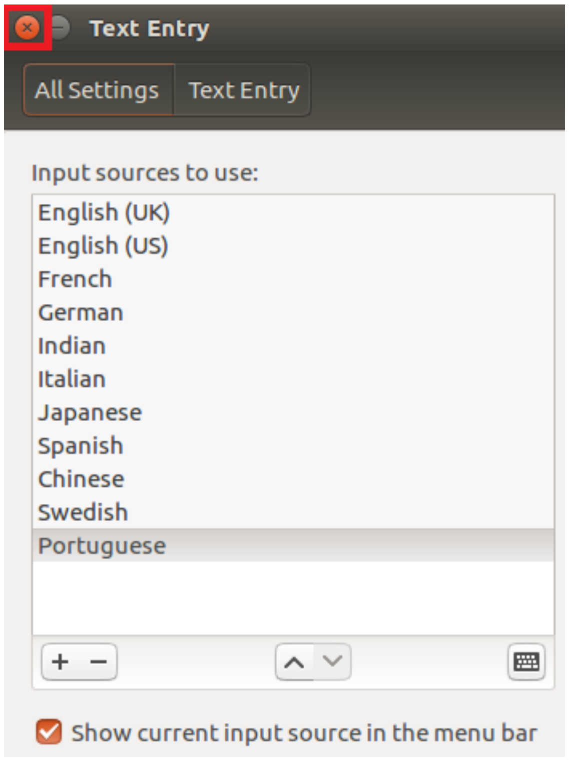


Select an input source to add

Esperanto (Portugal, Nativo)
Portuguese
Portuguese (Brazil)
Portuguese (Brazil, Dvorak)
Portuguese (Brazil, eliminate dead keys)
Portuguese (Brazil, Nativo)
Portuguese (Brazil, Nativo for US keyboards)
Portuguese (Brazil, Sun Type 6/7)
Portuguese (eliminate dead keys)
Portuguese (Macintosh)
Portuguese (Macintosh, eliminate dead keys)
Portuguese (Macintosh, Sun dead keys)

portu

__ **6.** **Close** the Settings box by clicking on the “**x**” in the top-right of the dialogue:

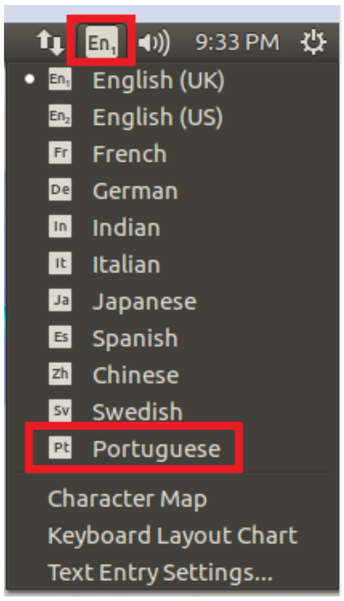


Input sources to use:

English (UK)
English (US)
French
German
Indian
Italian
Japanese
Spanish
Chinese
Swedish
Portuguese

Show current input source in the menu bar

__ **7.** Select the En₂ in the top right of the screen and select your new keyboard:



__ **8.** Your keyboard is now ready to use.

We Value Your Feedback!

- Your feedback is very important to us as we use it to continually improve the lab material.
- To give us feedback after the lab has finished, please send your comments to “**blockchain@uk.ibm.com**”