Contents

[1. File Overview 2](#_Toc457383474)

[2. Admin 3](#_Toc457383475)

[2.1 Demo 3](#_Toc457383476)

[2.2 Identity 6](#_Toc457383477)

[3. Blockchain 7](#_Toc457383478)

[3.1 Blocks 7](#_Toc457383479)

[3.2 Block 7](#_Toc457383480)

[4. Vehicles 9](#_Toc457383481)

[5. Vehicle 12](#_Toc457383482)

[5.2 Colour 13](#_Toc457383483)

[5.3 Make 14](#_Toc457383484)

[5.4 Model 16](#_Toc457383485)

[5.5 Owner 18](#_Toc457383486)

[5.6 Registration 20](#_Toc457383487)

[5.7 Scrap 21](#_Toc457383488)

[5.8 VIN 22](#_Toc457383489)

[6. Participants 25](#_Toc457383490)

[6.2 Regulators 26](#_Toc457383491)

[6.3 Manufacturers 27](#_Toc457383492)

[6.4 Dealerships 28](#_Toc457383493)

[6.5 Lease Companies 28](#_Toc457383494)

[6.6 Leaseese 29](#_Toc457383495)

[6.7 Scrap Merchants 30](#_Toc457383496)

[7. Transactions 31](#_Toc457383497)

[8. Chaincode 32](#_Toc457383498)

[8.1 Vehicles 32](#_Toc457383499)

# File Overview

This document defines the API that runs on the NodeJS server. For the Hyperledger Fabric API click [here](https://github.com/hyperledger/fabric/blob/master/docs/API/CoreAPI.md#rest-api)

# Admin

## Demo

### POST      /admin/demo

Type: POST  
Input Type: JSON  
Input Object: {"scenario":<scenario\_type>}  
Transfer Encoding: Chunked  
Response Type: Single  
Response Format: JSON Object  
Success: {"message": "performing scenario   
 creation now"}

##### Description

Creates a 3 or 10 vehicle scenario. It writes to a file what stage it is at and also write errors to this file. An error JSON object with a field error:true is one that caused the call to stop at that point. This data can be retrieved using GET /admin/demo.

##### Successful output to demo\_status.log:

{"message": "Creating vehicles"}&&  
{"message": "Created vehicle <v5c\_id>", counter":true}&&  
{"message":"Created vehicle <v5c\_id>", "counter":true}&&  
{"message":"Transferring vehicles to manufacturers"}&&  
{"message":"Transfered vehicle <v5c\_id>(<username> -> <username>)","counter":true}&&  
{"message":"Updating vehicles' details"}&&  
{"message":"Updated all fields for vehicle <v5c\_id>","counter":true}&&  
{"message":"Updated all fields for vehicle <v5c\_id>", "counter":true}&&  
{"message":"Transferring vehicles to private owners"}&&  
{"message":"Transfered all owners for vehicle <v5c\_id>","counter":true}&&  
{"message":"Demo setup"}

##### Error output to demo\_status.log:

Output: {"message": "Scenario type not

recognised", "error": true}

Description: The user sent an invalid scenario type.

Solutions:

Make sure the sent JSON contains a

field scenario with the value

"1\_vehicle", "3\_vehicle" or

"10\_vehicle".

Output: {"message": "Initial vehicles file not   
 found", error: true}

Description: Cannot find the initial vehicles JSON file.

Solutions:

1. Ensure that the initial vehicles JSON   
 file is at Server\_Side/blockchain/assets  
 /vehicles/initial\_vehicles.json

Output: {"message": "Unable to write vehicle",   
 "error": true}

Description: The app was unable to invoke the chaincode   
 to produce a new vehicle and query that the   
 vehicle was created.

Solutions:

1. Make sure that the chaincode is deployed   
 and that the name in the configuration   
 file is correct.

2. Make sure that the Blockchain network is   
 running.

3. Make sure the IP and Port in   
 configuration.js are correct.

Output: {"message": "Unable to transfer vehicles",   
 "error": true}

Description: The app was unable to invoke the chaincode   
 to transfer a vehicle's ownership and   
 confirm the change with a query.

Solutions:

1. Make sure the chaincode is deployed and   
 the name in the configuration file is   
 correct.

2. Make sure that the Blockchain network is   
 running.

3. Make sure that the transfer in the   
 intial\_vehicles.json is allowed by the   
 chaincode. For example a vehicle cannot   
 be owned by Alfa Romeo then immediately   
 after a Scrap Merchant.

Ouput: {"message": "Unable to update vehicles",   
 "error":true}

Description: The app was unable to invoke the chaincode   
 to update a vehicles field and confirm the   
 change with a query.

Solutions:

1. Make sure the chaincode is deployed and   
 the name in the configuration file is   
 correct.

2. Make sure that the Blockchain network is   
 running.

### GET        /admin/demo

Type: GET  
Transfer Encoding: Chunked  
Response Type: Single  
Response Format: JSON Objects Split by && Delimiter  
Success: <demo\_status.log>

##### Description

Returns the contents of the demo\_status.log file which is updated by running POST /admin/demo.

##### Errors

Output: {"message": "Unable to load demo\_status.log   
 file", "error": true}  
Status: 400  
Description:   
 1. Make sure that the demo\_status.log file   
 is in Server\_Side/logs

Output: {"message": "Invalid JSON Object", "error":   
 true}  
Status: 400  
Description:   
 1. Make sure that the demo\_status.log file   
 is valid JSON

## Identity

### POST      /admin/identity

Type: POST  
Input Type: JSON  
Input Object: {"account":<username>,   
 "participantType":<participant\_type>}  
Transfer Encoding: Chunked  
Response Type: Single  
Response Format: JSON Object  
Success: {"message": "successfully logged user   
 in"}

##### Description

Registers the user passed with the peer.

##### Errors

Output: {"message":"Unable to log user in"}  
Status: 400  
Description: The user was unable to be registered with the peer.  
Solutions:   
 1. Make sure the user has been created with   
 the CA. If not use POST /blockchain  
 /participants.  
 2. Make sure the user is in the   
 Server\_Side/blockchain/participants  
 /participants\_info.js file.  
 3. Make sure the Blockchain network is   
 running.  
 4. Make sure the IP and Port in   
 configuration.js are correct.

# Blockchain

## Blocks

### GET        /blockchain/blocks

Type: GET  
Transfer Encoding: Chunked  
Response Type: Single  
Response Format: JSON Object   
Success: {"height": <height>,   
 "currentBlockHash": <block\_hash>}

##### Description

Returns the height of the Blockchain and the hash of the last block in the chain.

##### Errors

Output: {"message": "Unable to get chain length",   
 "error": true}  
Status: 400  
Description: The length of the chain was unobtainable.  
Solutions:  
 1. Make sure the Blockchain network is   
 running.  
 2. Make sure the IP and Port in   
 configuration.js are correct.

## Block

### GET        /blockchain/blocks/<\_number>

Type: GET  
Transfer Encoding: Chunked  
Response Type: Single  
Response Format: JSON Object  
Sucess: {"block": <block>}

##### Description

Takes a number and returns the block data for the block at that position in the Blockchain.

##### Errors

Output: {"message": "Unable to get block", "error":   
 true}  
Status: 400  
Description: The block was unobtainable.  
Solutions:  
 1. Make sure the Blockchain network is   
 running.  
 2. Make sure the block requested exists in   
 the chain e.g. block 240 doesn't exist in   
 a chain that is only 239 blocks long.   
 Note that the height of the chain is 1   
 greater than the position of the last   
 block.   
 3. Make sure the IP and Port in   
 configuration.js are correct.

# Vehicles

### POST      /blockchain/assets/vehicles

Type: POST  
Transfer Encoding: Chunked  
Response Type: Streamed  
Response Format: JSON Objects Split by && Delimiter  
Success: {"message":"Generating V5cID"}&&  
 {"message":"Checking V5cID is unique"  
 }&&{"message":"Creating vehicle with   
 v5cID: <v5c\_ID>"}&&{"message":   
 "Achieving Consensus"}&&{"message":   
 "Creation confirmed", "v5cID":   
 <v5c\_ID>}

##### Description

Invokes the vehicle chaincode to store a new vehicle in the world state.

##### Errors

Output: {"message": "Timeout while checking v5cID is   
 unique", "error": true, "v5cID": <v5c\_ID>}

Status: 400

Description: A timeout occurred while the function tried   
 to query the chaincode to check whether the   
 v5cID it generated was unique.

Solutions:

1. Make sure the user making the request is   
 a regulator.

2. Make sure the CA is running and eCerts   
 are queryable.

Output: {"message": "Unable to confirm v5cID is   
 unique", "error": true, "v5cID": <v5c\_ID>}

Status: 400

Description: The function was unable to contact the   
 Blockchain network to query the generated   
 v5cID.

Solutions:

1. Make sure the Blockchain network is   
 running.

2. Make sure the IP and Port in   
 configuration.js are correct.

3. Make sure the chaincode is running.

4. Make sure the chaincode name in   
 configuration.js is correct.

Output: {"message": "Unable to create vehicle",   
 "error": true, "v5cID": <v5c\_ID>}

Status: 400

Description: The function was unable to contact the   
 Blockchain network to query the generated   
 v5cID.

Solutions:

1. Make sure the Blockchain network is   
 running.

2. Make sure the chaincode is running.

3. Make sure the chaincode name in   
 configuration.js is correct.

Output: {"message": "Unable to confirm vehicle   
 create. Request timed out.", "error": true,   
 "v5cID": <v5c\_ID>}

Status: 400

Description: The function timed out when trying to query   
 the newly created vehicle.

Solutions:

1. Make sure the chaincode is running.

2. Make sure the CA is running and eCerts   
 are queryable.

### GET        /blockchain/assets/vehicles

Type: GET

Transfer Encoding: Chunked

Response Type: Streamed

Response Format: JSON Objects Split by && Delimiter

Success: {"VIN": <vin>, "make": <string>,   
 "model": <string>, "reg": <string>,   
 "owner": <username>,

"colour": <string>,   
 "scrapped": <bool>,   
 "status": <int>, "v5cID": <v5c\_id>   
 }&&{"VIN": <vin>, "make": <string>,   
 "model": <string>, "reg": <string>,   
 "owner": <username>,

"colour": <colour>,   
 "scrapped": <bool>,   
 "status": <int>, "v5cID": <v5c\_id>   
 }&&...

##### Description

Queries the vehicle chaincode and returns all the vehicles that the current user in the session has permission to get details about.

##### Errors

Output: {"message": "Unable to get blockchain   
 assets", "error": "true"}  
Status: 400  
Descriptipon: The function was unable to query the   
 chaincode to retrieve the vehicle assets   
 owned by the current session's user.  
Solutions:  
 1. Make sure that the chaincode is running.  
 2. Make sure the chaincode name in   
 configuration.js is correct.

# Vehicle

### DELETE   /blockchain/assets/vehicles/<v5c\_ID>

Type: DELETE  
Transfer Encoding: Chunked  
Response Type: Streamed   
Response Format: JSON Object Split By && Delimiter  
Success: {"message": "Formatting request"}&&  
 {"message": "Updating scrap value"}&&  
 {"message": "Achieving consensus"}&&  
 {"message": "Scrap updated"}

##### Description

Invokes the vehicle chaincode function to update the scrapped field to true.

##### Errors

Output: {"message": "Unable to update scrap",   
 "error": true, "v5cID": <v5c\_ID>}

Status: 400

Description: The function was unable to invoke the   
 chaincode to scrap the vehicle.

Solutions:

1. Make sure the Blockchain network is   
 running.

2. Make sure the IP and Port in   
 configuration.js are correct.

Output: {"message": "Unable to confirm scrap update.   
 Request timed out.", "error": true, "v5cID": <v5c\_ID>}

Status: 400

Description: The function was unable to confirm that the   
 scrap was updated within the time limit by   
 querying the chaincode.

Solutions:

1. Make sure that the chaincode is running.

2. Make sure the chaincode name in   
 configuration.js is correct.

3. Make sure the user calling has permission   
 to perform a scrap update.

4. Make sure the <v5c\_ID> passed exists.

5. Make sure the CA is running and eCerts   
 are queryable.

## Colour

### GET        /blockchain/assets/vehicles/<v5c\_ID>/colour

Type: GET  
Transfer Encoding: Chunked  
Response Type: Single  
Response Format: JSON Object  
Success: {"message": <string>}

##### Description

Queries the vehicle chaincode and returns the colour of the vehicle.

### Errors

Output: {"message": "Unable to read colour",   
 "error": true, "v5cID": <v5c\_ID>}  
Status: 400  
Description: The function was unable to query the vehicle   
 chaincode to retrieve the vehicle's details.  
Solutions:  
 1. Make sure the Blockchain network is   
 running.  
 2. Make sure the IP and Port in   
 configuration.js are correct.  
 3. Make sure that the chaincode is running.  
 4. Make sure the chaincode name in   
 configuration.js is correct.  
 5. Make sure the user calling has permission   
 to read the vehicle.  
 6. Make sure the <v5c\_ID> passed exists.  
 7. Make sure the CA is running and eCerts   
 are queryable.

### PUT        /blockchain/assets/vehicles/<v5c\_ID>/colour

Type: PUT  
Input Type: JSON  
Input Object: {"value": <string>}  
Transfer Encoding: Chunked  
Response Type: Streamed   
Response Format: JSON Object Split By && Delimiter  
Success: {"message": "Formatting request"}&&  
 {"message": "Updating colour value"}&&

{"message": "Achieving consensus"}&&  
 {"message": "Colour updated"}

##### Description

Invokes the vehicle chaincode function to update the colour field to the value passed.

##### Errors

Output: {"message": "Unable to update colour",   
 "error": true, "v5cID": <v5c\_ID>}

Status: 400

Description: The function was unable to invoke the   
 chaincode to cause the colour to update.

Solutions:

1. Make sure the Blockchain network is   
 running.

2. Make sure the IP and Port in   
 configuration.js are correct.

Output: {"message": "Unable to confirm colour   
 update. Request timed out.", "error": true,   
 "v5cID": <v5c\_ID>}

Status: 400

Description: The function was unable to confirm that the   
 colour was updated within the time limit by   
 querying the chaincode using the GET colour   
 API.

Solutions:

1. Make sure that the chaincode is running.

2. Make sure the chaincode name in   
 configuration.js is correct.

3. Make sure the user calling has permission   
 to perform a colour update.

4. Make sure the <v5c\_ID> passed exists.

5. Make sure the CA is running and eCerts   
 are queryable.

## Make

### GET        /blockchain/assets/vehicles/<v5c\_ID>/make

Type: GET  
Transfer Encoding: Chunked  
Response Type: Single  
Response Format: JSON Object  
Success: {"message": <string>}

##### Description

Queries the vehicle chaincode and returns the make of the vehicle.

##### Errors

Output: {"message": "Unable to read make", "error":   
 true, "v5cID": <v5c\_ID>}  
Status: 400  
Description: The function was unable to query the vehicle   
 chaincode to retrieve the vehicle's details.  
Solutions:  
 1. Make sure the Blockchain network is   
 running.  
 2. Make sure the IP and Port in   
 configuration.js are correct.  
 3. Make sure that the chaincode is running.  
 4. Make sure the chaincode name in   
 configuration.js is correct.  
 5. Make sure the user calling has permission   
 to read the vehicle.  
 6. Make sure the <v5c\_ID> passed exists.  
 7. Make sure the CA is running and eCerts   
 are queryable.

### PUT        /blockchain/assets/vehicles/<v5c\_ID>/make

Type: PUT  
Input Type: JSON  
Input Object: {"value": <string>}  
Transfer Encoding: Chunked  
Response Type: Streamed   
Response Format: JSON Object Split By && Delimiter  
Success: {"message": "Formatting request"}&&  
 {"message": "Updating make value"}&&  
 {"message": "Achieving consensus"}&&  
 {"message": "Make updated"}

##### Description

Invokes the vehicle chaincode function to update the make field to the value passed.

##### Errors

Output: {"message": "Unable to update make",   
 "error": true, "v5cID": <v5c\_ID>}  
Status: 400  
Description: The function was unable to invoke the   
 chaincode to cause the make to update.  
Solutions:  
 1. Make sure the Blockchain network is   
 running.  
 2. Make sure the IP and Port in   
 configuration.js are correct.

Output: {"message": "Unable to confirm make update.   
 Request timed out.", "error": true, "v5cID":   
 <v5c\_ID>}

Status: 400

Description: The function was unable to confirm that the   
 make was updated within the time limit by   
 querying the chaincode using the GET make   
 API.

Solutions:

1. Make sure that the chaincode is running.

2. Make sure the chaincode name in   
 configuration.js is correct.

3. Make sure the user calling has permission   
 to perform a make update.

4. Make sure the <v5c\_ID> passed exists.

5. Make sure the CA is running and eCerts   
 are queryable.

## Model

### GET        /blockchain/assets/vehicles/<v5c\_ID>/model

Type: GET  
Transfer Encoding: Chunked  
Response Type: Single  
Response Format: JSON Object  
Success: {"message": <string>}

##### Description

Queries the vehicle chaincode and returns the model of the vehicle.

##### Errors

Output: {"message": "Unable to read model", "error":   
 true, "v5cID": <v5c\_ID>}  
Status: 400  
Description: The function was unable to query the vehicle   
 chaincode to retrieve the vehicle's details.  
Solutions:  
 1. Make sure the Blockchain network is   
 running.  
 2. Make sure the IP and Port in   
 configuration.js are correct.  
 3. Make sure that the chaincode is running.  
 4. Make sure the chaincode name in   
 configuration.js is correct.  
 5. Make sure the user calling has permission   
 to read the vehicle.  
 6. Make sure the <v5c\_ID> passed exists.  
 7. Make sure the CA is running and eCerts   
 are queryable.

### PUT        /blockchain/assets/vehicles/<v5c\_ID>/model

Type: PUT  
Input Type: JSON  
Input Object: {"value": <string>}  
Transfer Encoding: Chunked  
Response Type: Streamed   
Response Format: JSON Object Split By && Delimiter  
Success: {"message": "Formatting request"}&&  
 {"message": "Updating model value"}&&  
 {"message": "Achieving consensus"}&&  
 {"message": "Model updated"}

##### Description

Invokes the vehicle chaincode function to update the model field to the value passed.

##### Errors

Output: {"message": "Unable to update model",   
 "error": true, "v5cID": <v5c\_ID>}  
Status: 400  
Description: The function was unable to invoke the   
 chaincode to cause the model to update.  
Solutions:  
 1. Make sure the Blockchain network is   
 running.  
 2. Make sure the IP and Port in   
 configuration.js are correct.

Output: {"message": "Unable to confirm model update.   
 Request timed out.", "error": true, "v5cID":   
 <v5c\_ID>}

Status: 400

Description: The function was unable to confirm that the   
 model was updated within the time limit by   
 querying the chaincode using the GET model   
 API.

Solutions:

1. Make sure that the chaincode is running.

2. Make sure the chaincode name in   
 configuration.js is correct.

3. Make sure the user calling has permission   
 to perform a model update.

4. Make sure the <v5c\_ID> passed exists.

5. Make sure the CA is running and eCerts   
 are queryable.

## Owner

### GET        /blockchain/assets/vehicles/<v5c\_ID>/owner

Type: GET  
Transfer Encoding: Chunked  
Response Type: Single  
Response Format: JSON Object  
Success: {"message": <string>}

##### Description

Queries the vehicle chaincode and returns the owner of the vehicle.

##### Errors

Output: {"message": "Unable to read owner", "error":   
 true, "v5cID": <v5c\_ID>}  
Status: 400  
Description: The function was unable to query the vehicle   
 chaincode to retrieve the vehicle's details.  
Solutions:  
 1. Make sure the Blockchain network is   
 running.  
 2. Make sure the IP and Port in   
 configuration.js are correct.  
 3. Make sure that the chaincode is running.  
 4. Make sure the chaincode name in   
 configuration.js is correct.  
 5. Make sure the user calling has permission   
 to read the vehicle.  
 6. Make sure the <v5c\_ID> passed exists.  
 7. Make sure the CA is running and eCerts   
 are queryable.

#### PUT        /blockchain/assets/vehicles/<v5c\_ID>/owner

Type: PUT  
Input Type: JSON  
Input Object: {"value": <string>}  
Transfer Encoding: Chunked  
Response Type: Streamed   
Response Format: JSON Object Split By && Delimiter  
Success: {"message": "Formatting request"}&&{"message": "Updating owner value"}&&{"message": "Achieving consensus"}&&{"message": "Owner updated"}

##### Description

Invokes the vehicle chaincode function to update the owner field to the value passed.

##### Errors

Output: {"message": "Unable to update owner",   
 "error": true, "v5cID": <v5c\_ID>}  
Status: 400  
Description: The function was unable to invoke the   
 chaincode to cause the owner to update.  
Solutions:  
 1. Make sure the Blockchain network is   
 running.  
 2. Make sure the IP and Port in   
 configuration.js are correct.

Output: {"message": "Unable to confirm owner update. Request timed out.", "error": true, "v5cID": <v5c\_ID>}

Status: 400

Description: The function was unable to confirm that the   
 owner was updated within the time limit by   
 querying the chaincode using the GET owner   
 API.

Solutions:

1. Make sure that the chaincode is running.

2. Make sure the chaincode name in   
 configuration.js is correct.

3. Make sure the user calling has permission   
 to perform an owner update.

4. Make sure the <v5c\_ID> passed exists.

5. Make sure the CA is running and eCerts   
 are queryable.

## Registration

### GET        /blockchain/assets/vehicles/<v5c\_ID>/reg

Type: GET  
Transfer Encoding: Chunked  
Response Type: Single  
Response Format: JSON Object  
Success: {"message": <string>}

##### Description

Queries the vehicle chaincode and returns the registration of the vehicle.

##### Errors

Output: {"message": "Unable to read registration",   
 "error": true, "v5cID": <v5c\_ID>}  
Status: 400  
Description: The function was unable to query the vehicle

chaincode to retrieve the vehicle's details.  
Solutions:  
 1. Make sure the Blockchain network is   
 running.  
 2. Make sure the IP and Port in   
 configuration.js are correct.  
 3. Make sure that the chaincode is running.  
 4. Make sure the chaincode name in   
 configuration.js is correct.  
 5. Make sure the user calling has permission   
 to read the vehicle.  
 6. Make sure the <v5c\_ID> passed exists.  
 7. Make sure the CA is running and eCerts   
 are queryable.

### PUT        /blockchain/assets/vehicles/<v5c\_ID>/reg

Type: PUT  
Input Type: JSON  
Input Object: {"value": <string>}  
Transfer Encoding: Chunked  
Response Type: Streamed   
Response Format: JSON Object Split By && Delimiter  
Success: {"message": "Formatting request"}&&  
 {"message": "Updating registration   
 value"}&&  
 {"message": "Achieving consensus"}&&  
 {"message": "Registration updated"}

##### Description

Invokes the vehicle chaincode function to update the reg field to the value passed.

##### Errors

Output: {"message": "Unable to update registration",   
 "error": true, "v5cID": <v5c\_ID>}  
Status: 400  
Description: The function was unable to invoke the   
 chaincode to cause the reg to update.  
Solutions:  
 1. Make sure the Blockchain network is   
 running.  
 2. Make sure the IP and Port in   
 configuration.js are correct.

Output: {"message": "Unable to confirm registration   
 update. Request timed out.", "error": true,   
 "v5cID": <v5c\_ID>}

Status: 400

Description: The function was unable to confirm that the   
 reg was updated within the time limit by   
 querying the chaincode using the GET   
 registration API.

Solutions:

1. Make sure that the chaincode is running.

2. Make sure the chaincode name in   
 configuration.js is correct.

3. Make sure the user calling has permission   
 to perform an registration update.

4. Make sure the <v5c\_ID> passed exists.

5. Make sure the CA is running and eCerts   
 are queryable.

## Scrap

### GET        /blockchain/assets/vehicles/<v5c\_ID>/scrap

Type: GET  
Transfer Encoding: Chunked  
Response Type: Single  
Response Format: JSON Object  
Success: {"message": <string>}

##### Description

Queries the vehicle chaincode and returns the scrap boolean of the vehicle.

##### Errors

Output: {"message": "Unable to read scrap", "error":   
 true, "v5cID": <v5c\_ID>}  
Status: 400  
Description: The function was unable to query the vehicle   
 chaincode to retrieve the vehicle's details.  
Solutions:  
 1. Make sure the Blockchain network is   
 running.  
 2. Make sure the IP and Port in   
 configuration.js are correct.  
 3. Make sure that the chaincode is running.  
 4. Make sure the chaincode name in   
 configuration.js is correct.  
 5. Make sure the user calling has permission   
 to read the vehicle.  
 6. Make sure the <v5c\_ID> passed exists.  
 7. Make sure the CA is running and eCerts   
 are queryable.

## VIN

### GET        /blockchain/assets/vehicles/<v5c\_ID>/VIN

Type: GET  
Transfer Encoding: Chunked  
Response Type: Single  
Response Format: JSON Object  
Success: {"message": <string>}

##### Description

Queries the vehicle chaincode and returns the vin of the vehicle.

##### Errors

Output: {"message": "Unable to read VIN", "error":   
 true, "v5cID": <v5c\_ID>}  
Status: 400  
Description: The function was unable to query the vehicle   
 chaincode to retrieve the vehicle's details.  
Solutions:  
 1. Make sure the Blockchain network is   
 running.  
 2. Make sure the IP and Port in   
 configuration.js are correct.  
 3. Make sure that the chaincode is running.  
 4. Make sure the chaincode name in   
 configuration.js is correct.  
 5. Make sure the user calling has permission   
 to read the vehicle.  
 6. Make sure the <v5c\_ID> passed exists.  
 7. Make sure the CA is running and eCerts   
 are queryable.

### PUT        /blockchain/assets/vehicles/<v5c\_ID>/VIN

Type: PUT  
Input Type: JSON  
Input Object: {"value": <string>}  
Transfer Encoding: Chunked  
Response Type: Streamed   
Response Format: JSON Object Split By && Delimiter  
Success: {"message": "Formatting request"}&&  
 {"message": "Updating VIN value"}&&  
 {"message": "Achieving consensus"}&&  
 {"message": "VIN updated"}

##### Description

Invokes the vehicle chaincode function to update the VIN field to the value passed.

##### Errors

Output: {"message": "Unable to update VIN", "error":   
 true, "v5cID": <v5c\_ID>}  
Status: 400  
Description: The function was unable to invoke the   
 chaincode to cause the VIN to update.  
Solutions:  
 1. Make sure the Blockchain network is   
 running.  
 2. Make sure the IP and Port in   
 configuration.js are correct.

Output: {"message": "Unable to confirm VIN update.   
 Request timed out.", "error": true, "v5cID":   
 <v5c\_ID>}

Status: 400

Description: The function was unable to confirm that the   
 VIN was updated within the time limit by   
 querying the chaincode using the GET VIN   
 API.

Solutions:

1. Make sure that the chaincode is running.

2. Make sure the chaincode name in   
 configuration.js is correct.

3. Make sure the user calling has permission   
 to perform an VIN update.

4. Make sure the <v5c\_ID> passed exists.

5. Make sure the CA is running and eCerts   
 are queryable.

# Participants

### POST      /blockchain/participants

Type: POST  
Transfer Encoding: Chunked  
Input Type: JSON  
Input Object: {"username": <string>,   
 "affiliation": <affiliation>,  
 "role": <role>, "company": <string>,   
 "street\_name": <string>,   
 "city": <string>,  
 "postcode": <string>}  
Response Type: Single  
Response Format: JSON Object  
Success: {"message": "User creation successful",   
 "id": <string>, "secret": <string>}

##### Description

Takes a username, affiliation and role and creates a new user with that account on the CA, logs them in and writes their details to the participants\_info.js file and config.js

##### Error

Output: {"message": "Cannot register users before   
 the CA connector is setup", "error": true}  
Status: 400  
Description: The RPC Connection has not set up yet.  
Success:   
 1. Wait for the RPC connection to be set up.

Output: {"error":"Unable to log user in"}  
Status: 400  
Description: The user was unable to be registered with   
 the peer.  
Solutions:   
 1. Make sure the user has been created with the CA.

If not use POST /blockchain/participants.  
 2. Make sure the user is in the Server\_Side  
 /blockchain/participants

/participants\_info.js file.  
 3. Make sure the Blockchain network is   
 running.  
 4. Make sure the IP and Port in   
 configuration.js are correct.

### GET        /blockchain/participants

Type: GET  
Transfer Encoding: Chunked  
Response Type: Single  
Response Format: JSON Object  
Success: {"result": {  
 "regulators": [<participant>],   
 "manufacturers": [<participant>],  
 "dealerships": [<participant>],  
 "lease\_companies": [<participant>],  
 "leasees": [<participant>],  
 "scrap\_merchants": [<participant>]  
 }}

##### Description

Reads the participants\_info.js file and returns the JSON of participants in it.

##### Errors

Output: {"message": "Unable to retrieve   
 participants", "error": true}  
Status: 404  
Description: The function was unable to find the   
 participants information.  
Solutions:  
 1. Make sure that the participants\_info.js   
 file is in Server\_Side/blockchain  
 /participants.  
 2. Make sure that the participants\_info.js   
 file has an exported variable called   
 participants\_info that is a JSON Object.

## Regulators

### GET        /blockchain/participants/regulators

Type: GET  
Transfer Encoding: Chunked  
Response Type: Single  
Response Format: JSON Object  
Success: {"result": [<participant>]}

##### Description

Retrieves the participants that are identified as regulators from the participants\_info.js file.

##### Errors

Output: {"message": "Unable to retrieve regulators",   
 "error": true}  
Status: 404   
Description: The function was unable to find the   
 participants information.  
Solutions:   
 1. Make sure that the participants\_info.js   
 file is in Server\_Side/blockchain  
 /participants.  
 2. Make sure that the participants\_info.js   
 file has an exported variable called   
 participants\_info that is a JSON Object.  
 3. Make sure that the participants\_info   
 variable JSON object has a field   
 regulators.

## Manufacturers

### GET        /blockchain/participants/manufacturers

Type: GET  
Transfer Encoding: Chunked  
Response Type: Single  
Response Format: JSON Object  
Success: {"result": [<participant>]}

##### Description

Retrieves the participants that are identified as manufacturers from the participants\_info.js file.

##### Errors

Output: {"message": "Unable to retrieve   
 manufacturers","error": true}  
Status: 404   
Description: The function was unable to find the   
 participants information.  
Solutions:   
 1. Make sure that the participants\_info.js   
 file is in Server\_Side/blockchain  
 /participants.  
 2. Make sure that the participants\_info.js   
 file has an exported variable called   
 participants\_info that is a JSON Object.  
 3. Make sure that the participants\_info   
 variable JSON object has a field   
 manufacturers.

## Dealerships

### GET        /blockchain/participants/dealerships

Type: GET  
Transfer Encoding: Chunked  
Response Type: Single  
Response Format: JSON Object  
Success: {"result": [<participant>]}

##### Description

Retrieves the participants that are identified as dealerships from the participants\_info.js file.

##### Errors

Output: {"message": "Unable to retrieve dealerships", "error": true}  
Status: 404   
Description: The function was unable to find the participants information.  
Solutions:   
 1. Make sure that the participants\_info.js   
 file is in Server\_Side/blockchain  
 /participants.  
 2. Make sure that the participants\_info.js   
 file has an exported variable called   
 participants\_info that is a JSON Object.  
 3. Make sure that the participants\_info   
 variable JSON object has a field   
 dealerships.

## Lease Companies

### GET        /blockchain/participants/lease\_companies

Type: GET  
Transfer Encoding: Chunked  
Response Type: Single  
Response Format: JSON Object  
Success: {"result": [<participant>]}

##### Description

Retrieves the participants that are identified as lease\_companies from the participants\_info.js file.

##### Errors

Output: {"message": "Unable to retrieve   
 lease\_companies", "error": true}  
Status: 404   
Description: The function was unable to find the   
 participants information.  
Solutions:   
 1. Make sure that the participants\_info.js   
 file is in Server\_Side/blockchain  
 /participants.  
 2. Make sure that the participants\_info.js   
 file has an exported variable called   
 participants\_info that is a JSON Object.  
 3. Make sure that the participants\_info   
 variable JSON object has a field   
 lease\_companies.

## Leaseese

### GET        /blockchain/participants/leasees

Type: GET  
Transfer Encoding: Chunked  
Response Type: Single  
Response Format: JSON Object  
Success: {"result": [<participant>]}

##### Description

Retrieves the participants that are identified as leasees from the participants\_info.js file.

### Errors

Output: {"message": "Unable to retrieve leasees",   
 "error": true}  
Status: 404   
Description: The function was unable to find the   
 participants information.  
Solutions:   
 1. Make sure that the participants\_info.js   
 file is in Server\_Side/blockchain  
 /participants.  
 2. Make sure that the participants\_info.js   
 file has an exported variable called   
 participants\_info that is a JSON Object.  
 3. Make sure that the participants\_info   
 variable JSON object has a field leasees.

## Scrap Merchants

### GET        /blockchain/participants/scrap\_merchants

Type: GET  
Transfer Encoding: Chunked  
Response Type: Single  
Response Format: JSON Object  
Success: {"result": [<participant>]}

##### Description

Retrieves the participants that are identified as scrap\_merchants from the participants\_info.js file.

##### Errors

Output: {"message": "Unable to retrieve   
 scrap\_merchants", "error": true}  
Status: 404   
Description: The function was unable to find the   
 participants information.  
Solutions:   
 1. Make sure that the participants\_info.js   
 file is in Server\_Side/blockchain  
 /participants.  
 2. Make sure that the participants\_info.js   
 file has an exported variable called   
 participants\_info that is a JSON Object.  
 3. Make sure that the participants\_info   
 variable JSON object has a field   
 scrap\_merchants.

# Transactions

### GET        /blockchain/transactions

Type: GET  
Transfer Enconding: Chunked  
Response Type: Single  
Response Format: JSON Object  
Success: {"transactions": [<transaction>]}

##### Description

Returns a JSON Object contain an array of all the transactions the user has permission to see. For the regulator this is all transactions but for the other users it is only transactions that relate to a vehicle they own or have owned. In the case of other users it will only show transactions before and up to the point they transferred the vehicle on. They cannot see what happens to the vehicle after they transfer it.

##### Errors

Output: {"message": "Unable to get chain length",   
 "error": true}  
Status: 400  
Description: The length of the chain was unobtainable.  
Solutions:  
 1. Make sure the Blockchain network is   
 running.  
 2. Make sure the IP and Port in   
 configuration.js are correct.  
  
Output: {"message": "Unable to get block   
 <block\_number>", "error": true}  
Status: 400  
Description: The block was unobtainable.  
Solutions:  
 1. Make sure the Blockchain network is   
 running.  
 2. Make sure the IP and Port in   
 configuration.js are correct.

# Chaincode

## Vehicles

### POST      /blockchain/chaincode/vehicles

Type: POST  
Transfer Encoding: Chunked  
Response Type: Single  
Response Format: JSON Object  
Success: {"message": <chaincode\_name>}

##### Description

Deploys the vehicle chaincode.

##### Errors

Output: {"message": "Unable to deploy chaincode",   
 error: true}  
Status: 400  
Description: The vehicle chaincode was unable to be   
 deployed  
 1. Make sure the Blockchain network is   
 running.  
 2. Make sure the IP and Port in   
 configuration.js are correct.  
 3. Make sure the chaincode path in   
 configuration.js is correct.

Output: {"message": "Unable to write chaincode   
 deploy name to configuration file", error:   
 true}

Status: 400

Description: The name of the deployed chaincode was   
 unable to be written to Server\_Side  
 /configurations/configuration.js

Solutions:

1. Make sure that Server\_Side/configurations  
 /configuration.js exists.

# Glossary

affiliation

Defines the type of user they are. They can be a Regulator, Manufacturer, Dealership, Lease Company, Leasee, Scrap Merchant. This affilitation is then mapped to a number for use in HyperLedger.

block

A block object defined in the HyperLedger fabric protocol spec.

block\_hash

The hash of the contents of a block. Defined further in the HyperLedger fabric protocol spec.

block\_number

An integer value for the number of a block in the chain.

chaincode\_name

The id of the chaincode. It is returned when chaincode is deployed and is used to tell the API which chaincode you wish to invoke or query. It is a 128 character string.

demo\_status.log

A file that stores the output of the demo setup process. Located in Server\_Side/logs.

height

Integer value for the length of the blockchain. 1 greater than the last block number.

participant

A JSON object outlining a user. Formatted as:

{

"name": "<string>",

"identity": "<username>",

"password": "<string>",

"address\_line\_1": "<string>",

"address\_line\_2": "<string>",

"address\_line\_3": "<string>",

"address\_line\_4": "<username>",

"postcode": "<string>"

}

participant\_type

A string defining the type of a user can be regulators, manufacturers, dealerships, lease\_companies, leasees, scrap\_merchants. Used to get the participants password from the participants info file.

role

Defines the role of the user. All users are currently set to 1. Roles are the same as defined in HyperLedger.

scenario\_type

The scenario type defines which scenario will be deployed when the POST admin/demo API is called. The values for this can be “full” or “simple” full will be interpreted to create the 10 car scenario and simple the 3.

transaction

A transaction object as defined by HyperLedger but with the payload converted from base64 to plain text and the cert parsed. It has the added fields of caller (who created the transaction) and failed (did the transaction changed the world state).

username

The identity of a user on the Blockchain.

v5c\_ID

The unique identifier for a vehicle object. It is used as the key for the vehicles JSON object when it is written to the world state. It consists of 2 uppercase letters followed by 7 letters for example AB1234567.

VIN

15 character integer or 0 if the vehicle has yet to be defined.