

## **Lab: Explore the IBM Blockchain service on IBM Bluemix**

## Overview

In this lab, you use the IBM® Blockchain service on IBM® Bluemix to build on the car leasing demo that was introduced in the previous lab, “Transfer assets in a business network.”

If you completed the previous lab, you have already deployed the car leasing application to your account, which means you can skip Step 1 and reuse your existing application.

**Tip:** This lab shows some screen captures in the IBM Bluemix interface in the classic view. If you log in to Bluemix and want to work in the classic view, click the avatar in the upper right and select **Switch to Classic** at the bottom of the avatar window.

**Important:** Because the IBM Blockchain service is in beta, it might be temporarily unavailable or at capacity. If you experience problems in the lab when accessing the IBM Blockchain service dashboard, try accessing the dashboard later.

## Prerequisites

It's recommended that you use Firefox or Chrome web browsers.

You will need a [Bluemix account](#) to create the sample application.

## Step 1. Deploy and configure the sample application

If you completed the previous “Transfer assets in a business network” lab, you may skip ahead to Step 2 in this lab. To deploy the sample application:

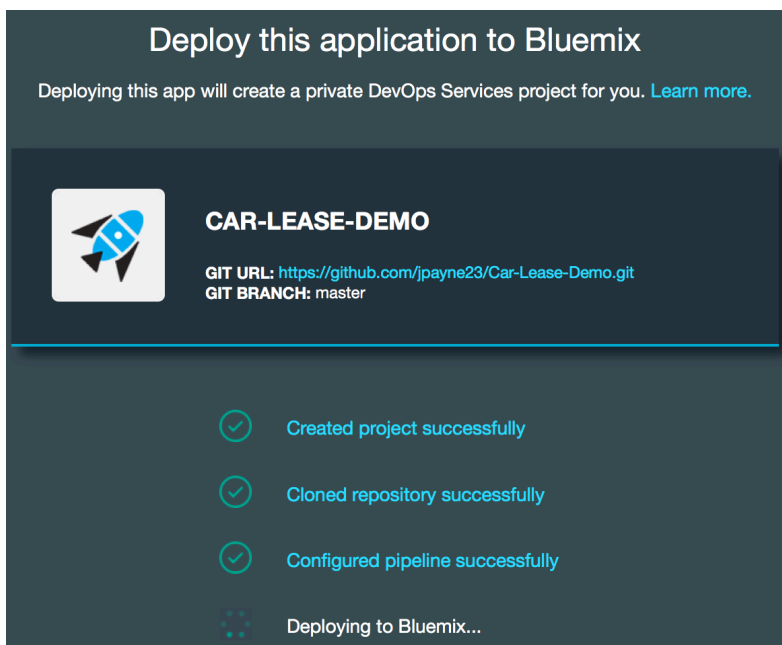
1. Open your browser and go to <http://www.bluemix.net>.
2. Click **Sign up** or **Log in** to create a new Bluemix account or log into your existing account.
3. After you have successfully signed up and logged into Bluemix, click **CATALOG** from the top bar.
4. Enter `Blockchain` in the search bar. When the service icon is displayed, click it to open the service information page.
5. Click **View Docs** to learn about the process of creating a blockchain environment.
6. Expand **Sample Apps and Tutorials** on the right side of the page to view the available apps.
7. Select the **Using Car Lease Demo** item from the list of apps
8. Click **Deploy to Bluemix** after the Car-Lease-Demo overview paragraph. You might need to log into Bluemix again.

If this is your first time using Bluemix DevOps services, you will be prompted to create an alias for the DevOps Services Git repository that will link to your IBM ID. This could be the first part of your email address (add a number afterward if needed to make it unique). Click **Create** after providing the alias.

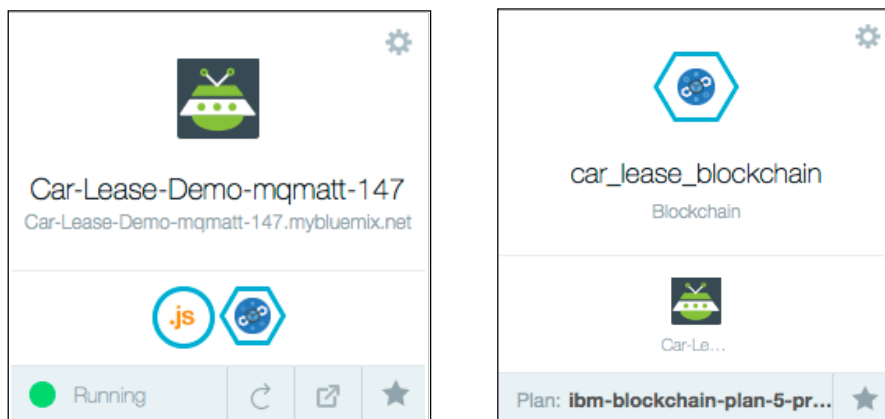


The screenshot shows a dialog box titled "IBM Bluemix™ DevOps Services" with the subtitle "Pick an alias". The text inside explains that a Git repository needs to be associated with an IBM ID via an alias, and defines an alias as a unique, publicly visible short name. There is a text input field with a placeholder "Pick an alias" and a speech bubble icon to its left. Below the input field is a checkbox labeled "I accept the DevOps Services [Terms of Use](#)". At the bottom is a blue "Create" button.

You can leave the `App Name`, `Region`, `Organization`, and `Space` attributes in the default state and click **Deploy**. It will take a few seconds for the default field values to be populated. This action will cause the Car-Lease-Demo to be deployed to your Bluemix environment and might take a couple of minutes to complete.



When you see the “Success!” message, click **DASHBOARD** to see the new car leasing application and associated blockchain service that you created.



9. Click the application icon in the dashboard.

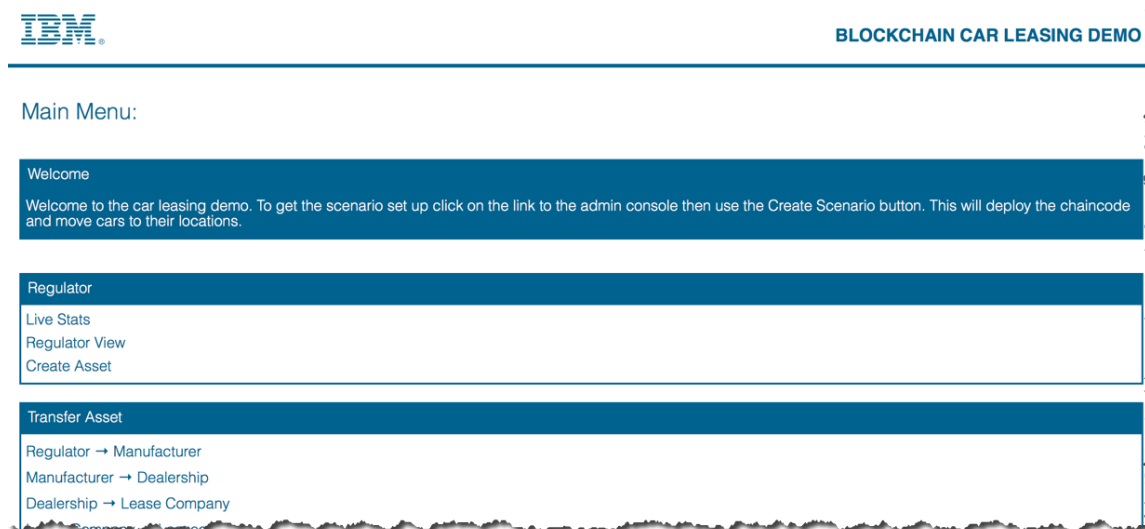


Your application icon might be different from the sample in the demo. Clicking the icon will show you information about the application, including memory consumption and the activity log.

10. To run the scenario, click the link displayed on the route for the application, for example:

Routes: [Car-Lease-Demo-mqmatt-147.mybluemix...](#)

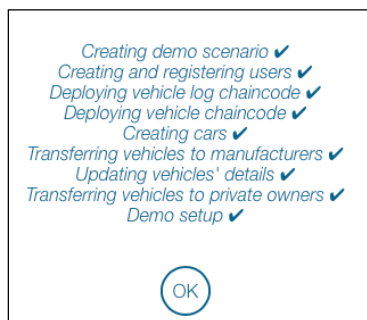
This will load a webpage that is served from the application.



11. Click **Admin Console > Create Full Scenario** to load the initial set of assets into the blockchain. This will take several minutes to complete.



The scenario setup is complete when “Demo setup” is displayed.




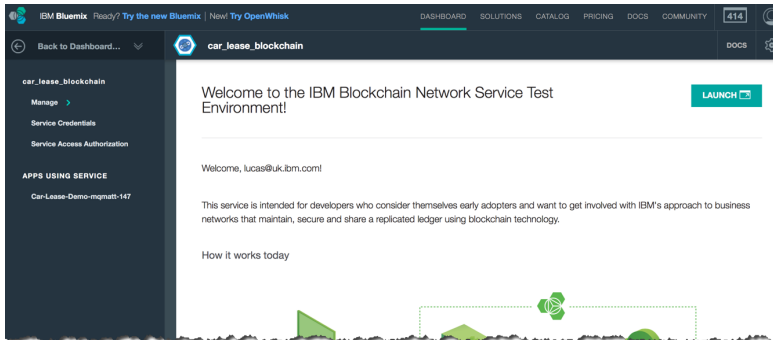
If an error occurs when creating the scenario, read “Remove the sample application” at the end of this document for instructions about how to delete the service.

## Step 2. Manage the sample application

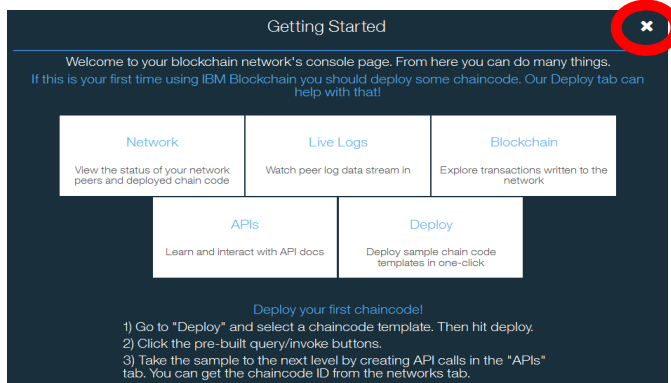
In this section, use the tools available in the IBM Bluemix environment to view and manage the blockchain.

### 2.1 View the components of the IBM Blockchain service

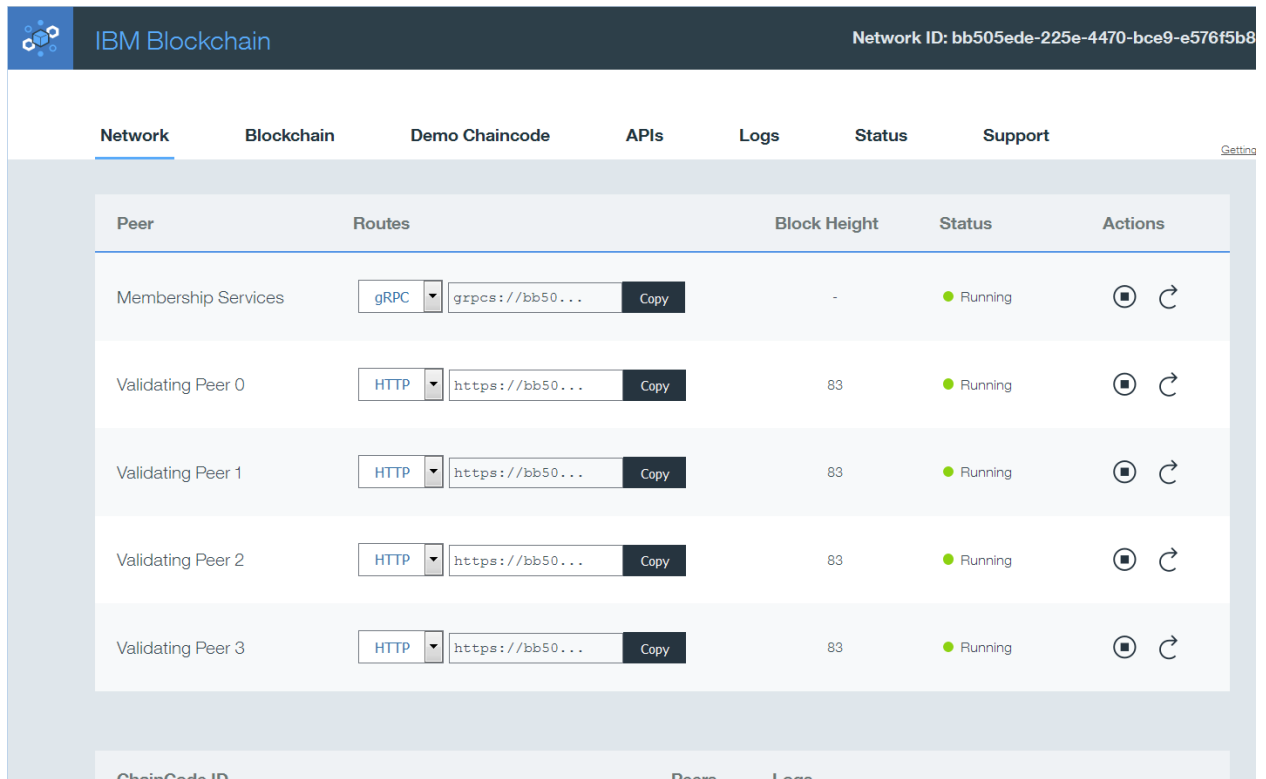
1. In Bluemix, click **DASHBOARD** to view the Car Leasing application.
2. Click the service icon for your new IBM Blockchain service in the Dashboard . This will take you to the service welcome page.



3. Review the details and click **LAUNCH** to start the service console.
4. Close the window that shows information about the sections. You will be able to look at these in more detail throughout this lab.



After closing the window, you will see the monitor page with the Network tab selected.



IBM Blockchain

Network ID: bb505ede-225e-4470-bce9-e576f5b8

Network Blockchain Demo Chaincode APIs Logs Status Support

Peer	Routes	Block Height	Status	Actions
Membership Services	<div>gRPC</div> <div>grpc://bb50...</div> <div>Copy</div>	-	Running	<div>Stop</div> <div>Refresh</div>
Validating Peer 0	<div>HTTP</div> <div>https://bb50...</div> <div>Copy</div>	83	Running	<div>Stop</div> <div>Refresh</div>
Validating Peer 1	<div>HTTP</div> <div>https://bb50...</div> <div>Copy</div>	83	Running	<div>Stop</div> <div>Refresh</div>
Validating Peer 2	<div>HTTP</div> <div>https://bb50...</div> <div>Copy</div>	83	Running	<div>Stop</div> <div>Refresh</div>
Validating Peer 3	<div>HTTP</div> <div>https://bb50...</div> <div>Copy</div>	83	Running	<div>Stop</div> <div>Refresh</div>

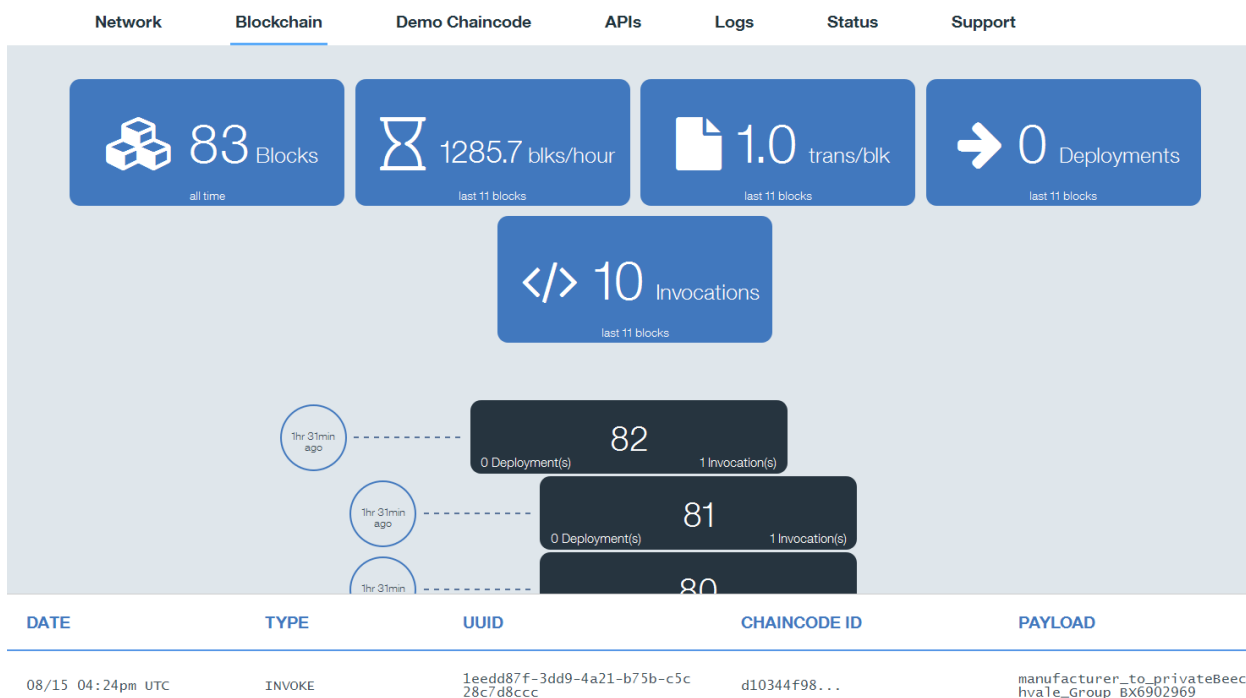
ChainCode ID Peers Logs

This view confirms that four validating peers and a certificate authority are running under the service that you created.

## 2.2 View the blockchain explorer

The blockchain explorer is a visual representation of the state of the blockchain.

1. Click the **Blockchain** tab at the top of the page.



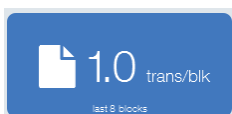
The icons show the following information:



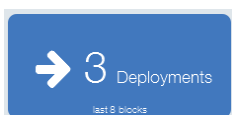
Total number of blocks in the chain.



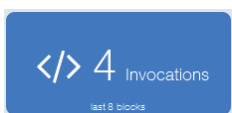
Average number of blocks per hour.



Number of transactions per block.



Number of deployment calls made to deploy chaincode.



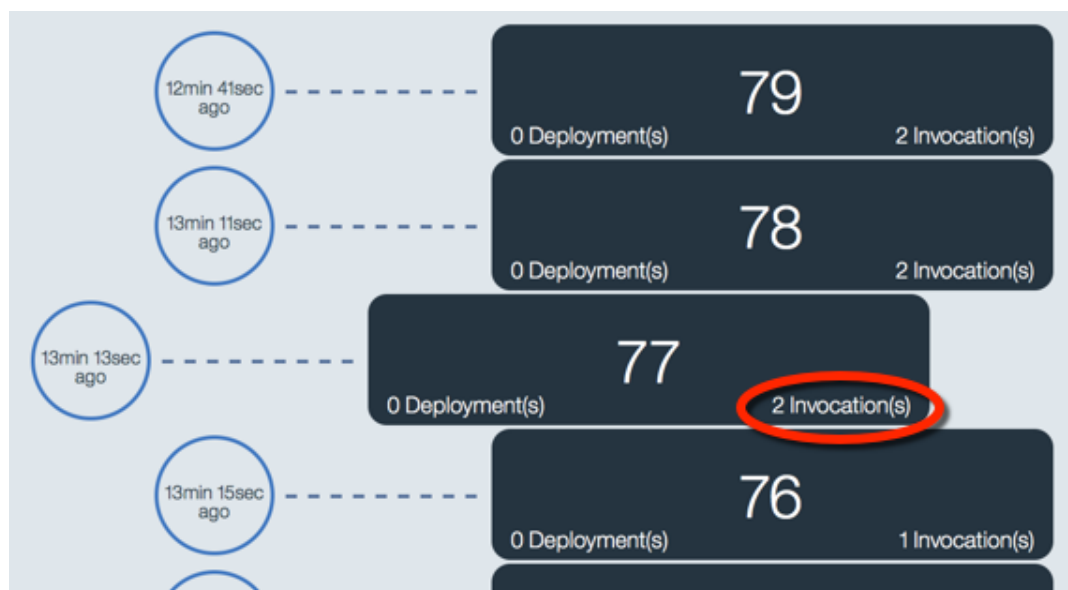
Number of invoke requests made within this blockchain.



Each block contains a set of transactions. In Hyperledger Fabric, a transaction is the record of the request to interact with chaincode, a smart contract. Two important transaction types are:

- **INVOKE:** The request to invoke a piece of chaincode, such as invoking the chaincode to transfer the ownership of a car.
- **DEPLOY:** The request to deploy a piece of chaincode across all validating peers so that it can be executed at a later date.

Other request types exist, such as QUERY, UPDATE, and TERMINATE. Not all request types are recorded on the blockchain.



The blocks also include when that block was committed to the blockchain.

2. Click a block that contains at least one invocation request.
3. Look through the list of transactions that are contained in the block.

DATE	TYPE	UUID	CHAINCODE ID	PAYLOAD
05/26 01:49pm UTC	INVOKE	47897ad9-4356-4b3c-9405-38856383a5ee	9c48cd2b3...	create_vehicle_logTransferELeaseCan & Joe Payne&[720965981630055]?toyota Yaris, Red, QD65 YKR DP8881!3826/5/2016 13:49:04LeaseCan Joe iayne

Each line of information is a transaction stored in the block. A block can contain multiple transactions, but in this demo, there will often be only one transaction per block because of the low frequency of transactions being made. The information being provided is:

- **Date:** The date the transaction was submitted.
- **Type:** The type of transaction taking place, such as INVOKE or DEPLOY.
- **UUID:** The unique identifier for each transaction.
- **Chaincode ID:** Refers to the chaincode that is being invoked or deployed.
- **Payload:** The input parameters of the chaincode.

4. Repeat this for other blocks to understand how the transactions are stored.

When the blockchain is initialized for the car leasing application, the first two blocks in the chain usually contain DEPLOY transactions by which the chaincode is deployed to the validating peers.

You can view these blocks if you're willing to scroll down the blockchain explorer that far!

## 2.3 Understand the blockchain peers

Now, you will review the logs associated with the peers. This is useful for understanding how the blockchain works and for diagnosing problems.

There are two ways of accessing the logs of the peers:

- The **Logs** button under the **Network** tab is useful for downloading log files from the peers for offline analysis.
- The **Live Logs** tab shows you what the peers are doing currently.

1. Click the **Network** tab on the service page.

Network

Blockchain

Demo Chaincode

APIs

Logs

Status

Support

Peer	Routes	Block Height	Status	Action
Membership Services	<div>gRPC</div> <div>grpc://bb50...</div> <div>Copy</div>	-	Running	
Validating Peer 0	<div>HTTP</div> <div>https://bb50...</div> <div>Copy</div>	83	Running	
Validating Peer 1	<div>HTTP</div> <div>https://bb50...</div> <div>Copy</div>	83	Running	
Validating Peer 2	<div>HTTP</div> <div>https://bb50...</div> <div>Copy</div>	83	Running	
Validating Peer 3	<div>HTTP</div> <div>https://bb50...</div> <div>Copy</div>	83	Running	

ChainCode ID

Peers

Logs

d10344f981cbdb01eaff5beba9bcf144d7ca8c54c...

Copy

4

VPO

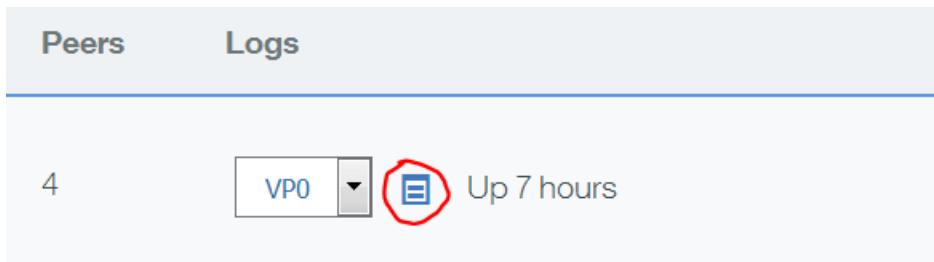
Up 7 hours

Here you can see that this blockchain network contains four validating peers and a Certificate Authority. The table underneath shows that there is one chaincode application deployed to this network.

Requests to invoke chaincode, including the method name and any input parameters, are replicated onto every validating node and when a block is created, every validating node will execute the chaincode independently. The validating peers will then attempt to achieve consensus over any changes proposed to the world state as a result of running this chaincode and as a consequence, will persist or discard the changes.

By looking at the logs for each peer, you can verify that every node has executed every transaction.

- Choose a chaincode ID from the table and click the log icon for the peer whose logs you want to examine.



A new tab opens with the logs for the selected peer.

- After you view the messages, close the browser tab.

```
ERR - 16:01:33.566 [shim] DEBU : Peer address: bb505ede-225e-4470-bce9-e576f5b837d3_vp1.us.blockchain.ibm.com:30
ERR - 16:01:33.633 [shim] DEBU : os.Args returns: [/opt/gopath
/bin/d10344f981cbb01eaff5bebafbcf144d7ca8c54c8c051d2b2e940281969ca900831fdf4e53017aa6a66d60d99e59b653d1f8eb6d8e
-peer.address=bb505ede-225e-4470-bce9-e576f5b837d3_vp1.us.blockchain.ibm.com:30303]
ERR - 16:01:33.633 [shim] DEBU : Registering.. sending REGISTER
ERR - 16:01:33.634 [shim] DEBU : []Received message REGISTERED from shim
ERR - 16:01:33.634 [shim] DEBU : []Handling ChaincodeMessage of type: REGISTERED(state:created)
ERR - 16:01:33.635 [shim] DEBU : Received REGISTERED, ready for invocations
ERR - 16:01:58.118 [shim] DEBU : [d10344f9]Received message INIT from shim
ERR - 16:01:58.118 [shim] DEBU : [d10344f9]Handling ChaincodeMessage of type: INIT(state:established)
ERR - 16:01:58.118 [shim] DEBU : Entered state init
ERR - 16:01:58.118 [shim] DEBU : [d10344f9]Received INIT, initializing chaincode
ERR - 16:01:58.119 [shim] DEBU : [d10344f9]Inside putstate, isTransaction = true
ERR - 16:01:58.119 [shim] DEBU : [d10344f9]Sending PUT_STATE
ERR - 16:01:58.119 [shim] DEBU : [d10344f9]Received message RESPONSE from shim
ERR - 16:01:58.119 [shim] DEBU : [d10344f9]Handling ChaincodeMessage of type: RESPONSE(state:init)
ERR - 16:01:58.119 [shim] DEBU : [d10344f9]before send
ERR - 16:01:58.119 [shim] DEBU : [d10344f9]after send
ERR - 16:01:58.119 [shim] DEBU : [d10344f9]Received RESPONSE, communicated (state:init)
ERR - 16:01:58.119 [shim] DEBU : [d10344f9]Received RESPONSE. Successfully updated state
ERR - 16:01:58.119 [shim] DEBU : [d10344f9]Inside putstate, isTransaction = true
ERR - 16:01:58.119 [shim] DEBU : [d10344f9]Sending PUT_STATE
ERR - 16:01:58.120 [shim] DEBU : [d10344f9]Received message RESPONSE from shim
ERR - 16:01:58.120 [shim] DEBU : [d10344f9]Handling ChaincodeMessage of type: RESPONSE(state:init)
ERR - 16:01:58.120 [shim] DEBU : [d10344f9]before send
ERR - 16:01:58.120 [shim] DEBU : [d10344f9]after send
ERR - 16:01:58.120 [shim] DEBU : [d10344f9]Received RESPONSE, communicated (state:init)
ERR - 16:01:58.120 [shim] DEBU : [d10344f9]Received RESPONSE. Successfully updated state
ERR - 16:01:58.120 [shim] DEBU : [d10344f9]Inside putstate, isTransaction = true
ERR - 16:01:58.120 [shim] DEBU : [d10344f9]Sending PUT_STATE
ERR - 16:01:58.120 [shim] DEBU : [d10344f9]Received message RESPONSE from shim
ERR - 16:01:58.120 [shim] DEBU : [d10344f9]Handling ChaincodeMessage of type: RESPONSE(state:init)
ERR - 16:01:58.120 [shim] DEBU : [d10344f9]before send
ERR - 16:01:58.120 [shim] DEBU : [d10344f9]after send
ERR - 16:01:58.120 [shim] DEBU : [d10344f9]Received RESPONSE, communicated (state:init)
ERR - 16:01:58.120 [shim] DEBU : [d10344f9]Received RESPONSE. Successfully updated state
ERR - 16:01:58.121 [shim] DEBU : [d10344f9]Inside putstate, isTransaction = true
```

## 2.4 Interact with the peers

You can invoke the management APIs that interact directly with the peers. In this lab, you'll be trying out these APIs directly from the IBM Blockchain service environment.

These APIs are used to operationally manage the blockchain. This is not the same as adding and invoking transactions through chaincode.

1. Click the **APIs** tab on the Service page.

Network Blockchain Demo Chaincode **APIs** Logs Status Support

Welcome to our Peer's API Swagger documentation.  
You can use this page to interact with your peers using their REST interface.

[https://bb505ede-225e-4470-bce9-e576f5b837d3\\_vp1.us.blockchain.ibm.com:443](https://bb505ede-225e-4470-bce9-e576f5b837d3_vp1.us.blockchain.ibm.com:443) VP0 VP1  
VP2 VP3

(feel free to change the peer url above)

[+ Network's Enroll IDs](#)

### IBM Blockchain API

Interact with the enterprise blockchain through IBM Blockchain API

Block	Show/Hide	List Operations	Expand Operations
Blockchain	Show/Hide	List Operations	Expand Operations
Chaincode	Show/Hide	List Operations	Expand Operations
Network	Show/Hide	List Operations	Expand Operations

This page allows you to invoke APIs that will directly interrogate and manage the blockchain. First, you will use the API endpoint to query the height of the blockchain (the number of blocks).

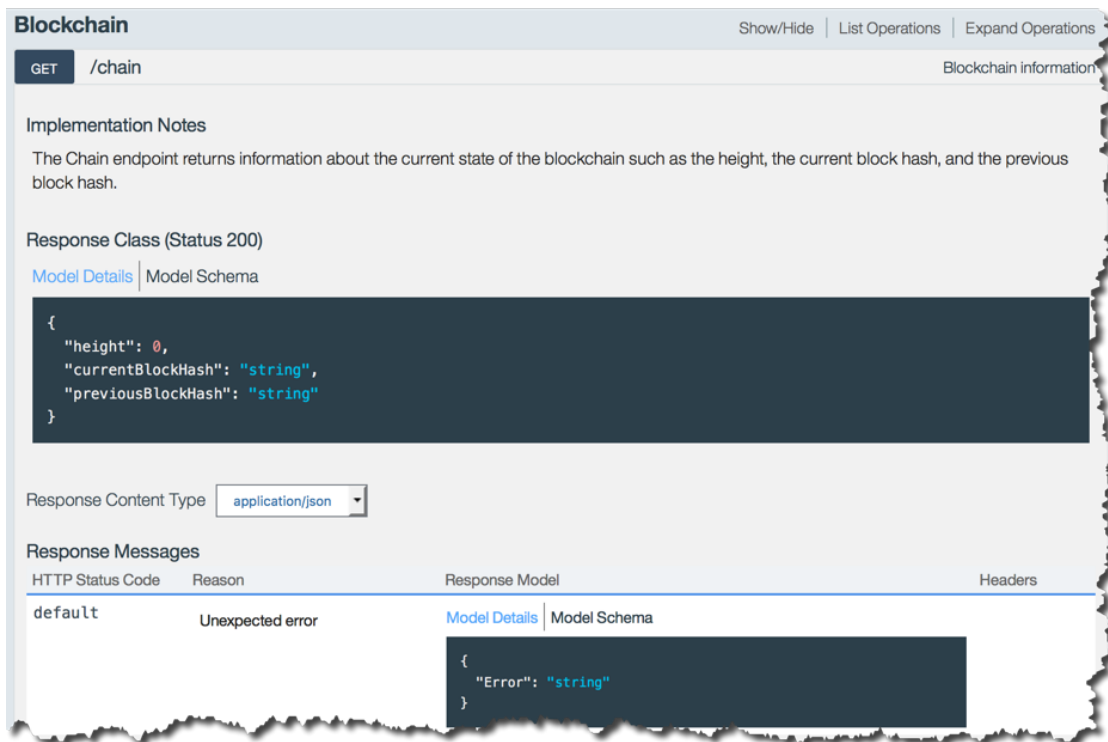
2. Click the **Blockchain** section.

This reveals the `GET /chain` operation, which is a valid method to call on the peer.

## Blockchain

**GET** `/chain`

3. Click **Expand Operations** to view information about this API. This displays the input and output data formats.



**Blockchain** Show/Hide List Operations Expand Operations

**GET** `/chain` Blockchain information

**Implementation Notes**

The Chain endpoint returns information about the current state of the blockchain such as the height, the current block hash, and the previous block hash.

**Response Class (Status 200)**

[Model Details](#) | [Model Schema](#)

```
{
  "height": 0,
  "currentBlockHash": "string",
  "previousBlockHash": "string"
}
```

Response Content Type:

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
default	Unexpected error	<a href="#">Model Details</a>   <a href="#">Model Schema</a>	

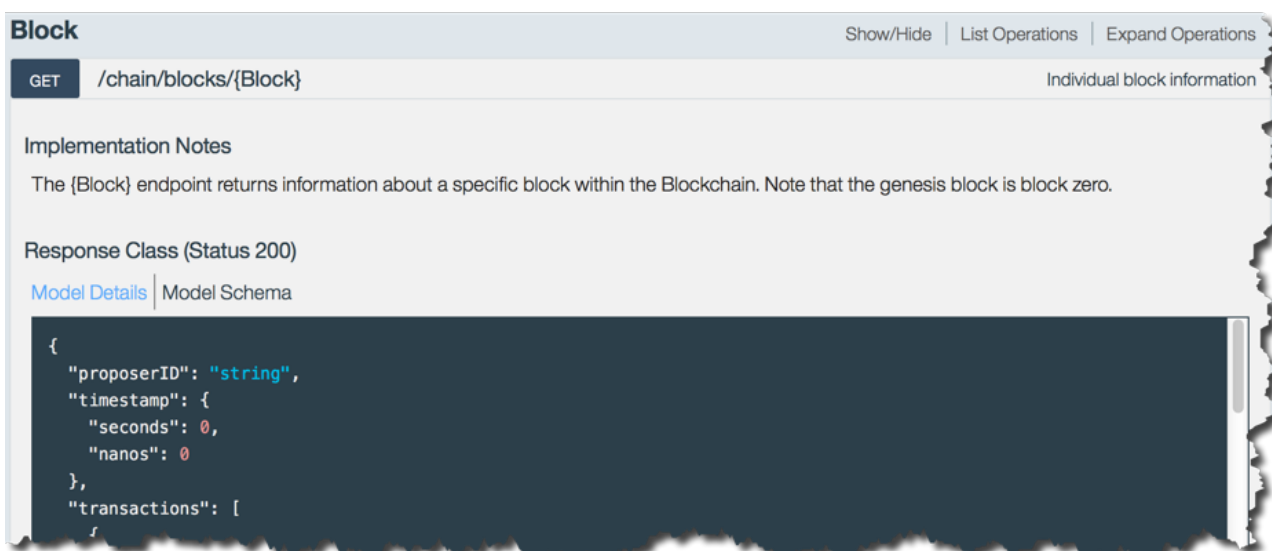
```
{
  "Error": "string"
}
```

- Click **Try It Out!** to invoke the API.



Review the displayed fields:

- **Request URL:** shows the URL that was invoked, including the endpoint information of the peer (hostname:port) and the method call (`/chain`).
  - **Response Body:** shows the information that was returned including, importantly, the height of the blockchain.
  - **Response Code:** 200 shows that the request was successful.
  - **Response Headers:** a field that confirms that the response body data was returned in a JSON data structure.
- Expand the **Block** section and review the information about how to interrogate an individual block in the blockchain.



- Enter the Block parameter to be a number less than the height of the chain and click **Try it out!**

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Block	40	Block number to retrieve	path	integer

Response Messages

HTTP Status Code	Reason	Response Model	Headers
default	Unexpected error	<a href="#">Model Details</a>   <a href="#">Model Schema</a> <pre> {   "Error": "string" } </pre>	

Try it out!

- Review the information returned in the **Response Body**.

Request URL

https://79a7456b-58ed-4bfb-9037-5c05f34fdbba\_vp1-api.blockchain.ibm.com:443/chain/blocks/40

Response Body

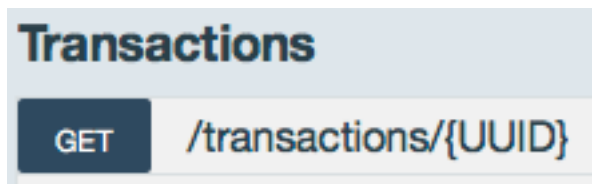
```

{
  "transactions": [
    {
      "type": 2,
      "chaincodeID": "EoABYzZIMWY5NjM3N2FhYzU5YmIwOTg1YjIyZWQ3OGE4OTU4ZGNkY2E3ZGhNzAyZmY0ODgyZmEzOTRiZTAzMDU4YmQ3OTE5MmIwYjY1ZmY4MmNkZGZjYThlMDA3MGEzNjA2OGQ0NDE",
      "payload": "CrlBCAESgwESgAFjNmUxZjk2Mzc3YWVJNTIYjA5ODVIMjZhdDc4Y1g5NThkY2RjYU3MDUjZjQ4ODUjYTM5NGJlMDMwNThiZDc5MTkyYjFiNjVjZjgyY2RkZmNhOGIwMDowYTM2MDY4ZDQ0",
      "uuid": "a338564e-ceef-4df6-9efd-95b65fa43efc",
      "timestamp": {
        "seconds": 1464270464,
        "nanos": 266423527
      },
      "nonce": "s+mSVUX6XeUBLf4br14YPp4sz56sXo",
      "cert": "MICJTCCAJOgAwIBAgIRAO011VhHEMIiZTZW6Aq+AUwCgYIKoZiZjOEAwMwKTElMAKGA1UEBHMCMVVMxODAKBgNVBAoTA0ICTEMMAoGA1UEAxMDdGNhMB4XDTE2MDU4YmQ3OTE5MmIwYjY1ZmY4MmNkZGZjYThlMDA3MGEzNjA2OGQ0NDE",
      "signature": "MEQCIB4J8gMyGZY52Bwzp2i2WeShrGUYKhvPwMgoQc3zB24AIBDatRcqdyOMBHyuOeXQBAADjNF+6ZBuDWqrp1RyIrgw=="
    }
  ],
  "stateHash": "0zUFHgcVkoNafxvRAdqMXNOETiZRoy4aDP1gxx2WBHENy+DPKZeBskSePwiYNIHzYINsEXtZLXmcl9g29elg==",
  "previousBlockHash": "hrCELkOTf5Xm0Rj6R2fnRyo/B2EIL3Nk64xZJ6TKPGqcgGcskVnjpsMXuKZ/17o2e00'AuqFrGz9'RBoknRw==s"
}

```



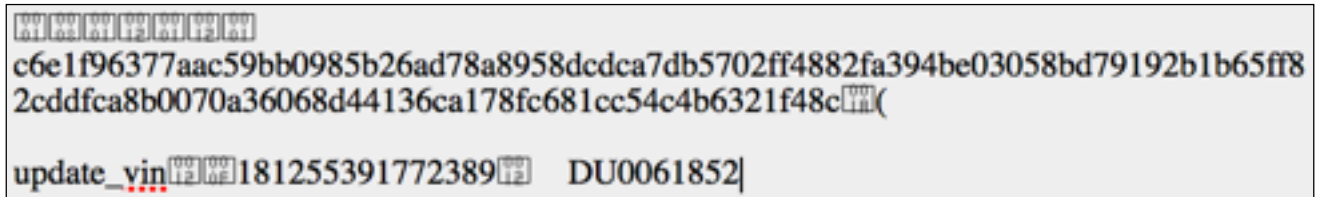
8. Copy the UUID field of a transaction from a block; this will be of the form a338564e-ceef-4df6-9efd-95b65fa43efc.
9. Click the **Transactions** section.



This reveals the GET /transactions/{UUID} operation, which is a valid method to call on the peer.

10. Paste the transaction UUID and click **Try it out!**

The **payload** field is base64 encoded. You can use a web tool such as <http://www.base64decode.org> to decode this information. When the information is decoded, you'll see that the payload includes the chaincode ID of the smart contract being called together with its input parameters as shown in the image.



This application does not encrypt the transactions, so the payloads are visible to all although they are encoded in base64.

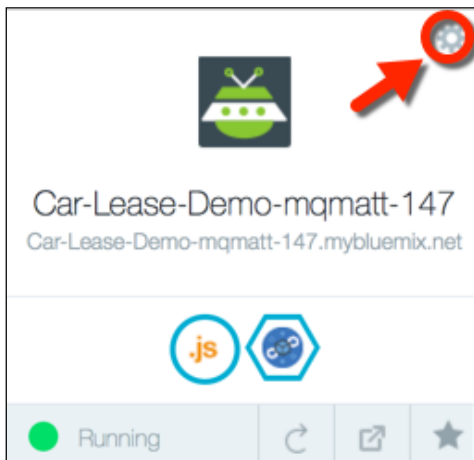
To be able to copy the full payload data, use the browser debug mode by pressing F12 or copy the Request URL shown and use cURL or Postman to make the request.

11. Interact with the other APIs that are available to you.

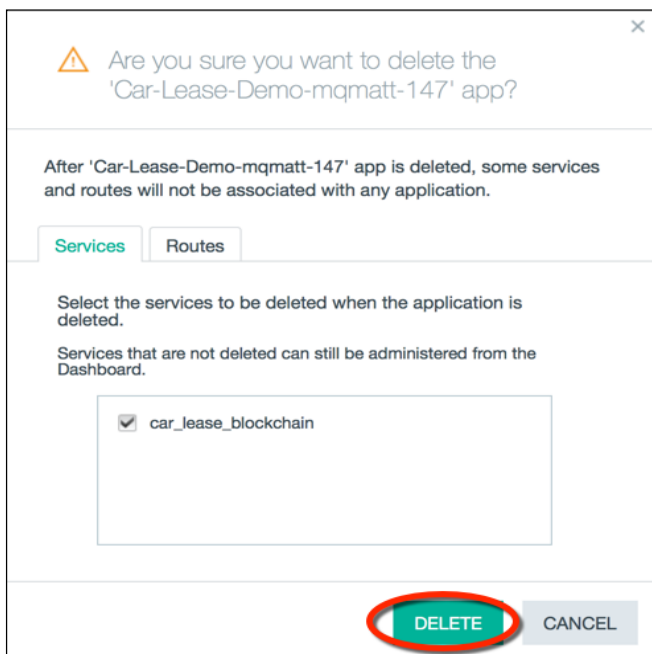
## Supplemental: Remove the sample application

If the blockchain service that you created crashes, here's how you can stop and remove it.

1. Click **Dashboard** to return to the Bluemix dashboard.
2. Click the settings icon in the upper-right corner of the car lease demo application.



3. Select **Delete App** from the menu.
4. Ensure that the `car_lease_blockchain` service is also selected for deletion and click **Delete**.



Wait for the items to stop and be deleted. After this is done, both the application and the associated service will no longer be visible in the Bluemix dashboard.