Modernizing enterprise business

Connecting application data with IBM Blockchain

IBM Oracle International Competency Center

Arun Sar
Krishna Harsha Voora
Mithun Hallikerehundi
Srirama Sharma
Yash Agrawal
Thasmiya Noorain A N

IBM Cognitive Systems

February 2019

Table of contents

Abstract

This paper provides guidance on how to establish connectivity between Hyperledger Fabric and Oracle Database using the node-oracledb and cx_Oracle modules and thus induce a sense of trust and reliability while modernizing enterprise businesses.

Introduction

Supply chain management involves multiple business stakeholders who are distributed geographically. Large volumes of transactions are carried out between participants of supply chain and these transactions need to be traceable, immutable, reliable, shared, and run with a sense of trust.

These transactions currently use applications connecting different parties or interact offline through emails. But, this is inefficient as there is considerable delay from different parties to respond. With IBM® Blockchain technology, we can have these complex transactions run on a blockchain network and save them in a chain of blocks. As each transaction occurs, it is put in a block and each block is connected to one before and after it. Transactions are blocked together, creating an irreversible chain that is distributed, permissible, and secure. Hyperledger Fabric is emerging as the widely accepted standard for enterprise blockchain platforms. Through open source and open governance, it features innovative new capabilities hardened for use by businesses, ushering in a new era of trust, transparency, and accountability.

This paper discusses a use case from supply chain management and explains how business operations are run on a blockchain network by streamlining portions of ERP and thus eventually decrease the turnaround time. This paper can be used as a reference to solve similar problems and could be integrated with enterprise applications.

Prerequisites

Oracle Database can run on IBM Power Systems™ as well as on IBM z Systems®. The IBM Blockchain Platform offers an IBM LinuxONE infrastructure option specifically optimized for blockchain and is integrated into IBM Cloud™ data centers. We have validated the stack on IBM Power Systems. To complete the steps in this paper, as a prerequisite, you need access to the following hardware and software:

- IBM® Power Systems™ (IBM POWER8® or IBM POWER9™)
- IBM AIX® 7.2
- Oracle Database 12.2
- Ubuntu 16.04 LTS
- Hyperledger Fabric v1.1.0
- node-oracledb
- cx_Oracle
- Node.js v8.11.2

Scenario description

This section briefs a scenario with the following problem:

- When a purchase order (PO) is created by a retailer, the PO could contain multiple items with different quantities for different suppliers.
- If the requested quantity or item is not available, the supplier contacts the retailer offline and the retailer updates the PO accordingly.

The following solution using Hyperledger Fabric has been proposed for the problem:

- The retailer can directly send a request to the blockchain network consisting of suppliers and respective suppliers could respond back to the retailer's request.
- This is carried out in a trustful manner using smart contracts to raise, validate, and respond to requests. The response is sent to the retailer in real time through events which can be handled accordingly.

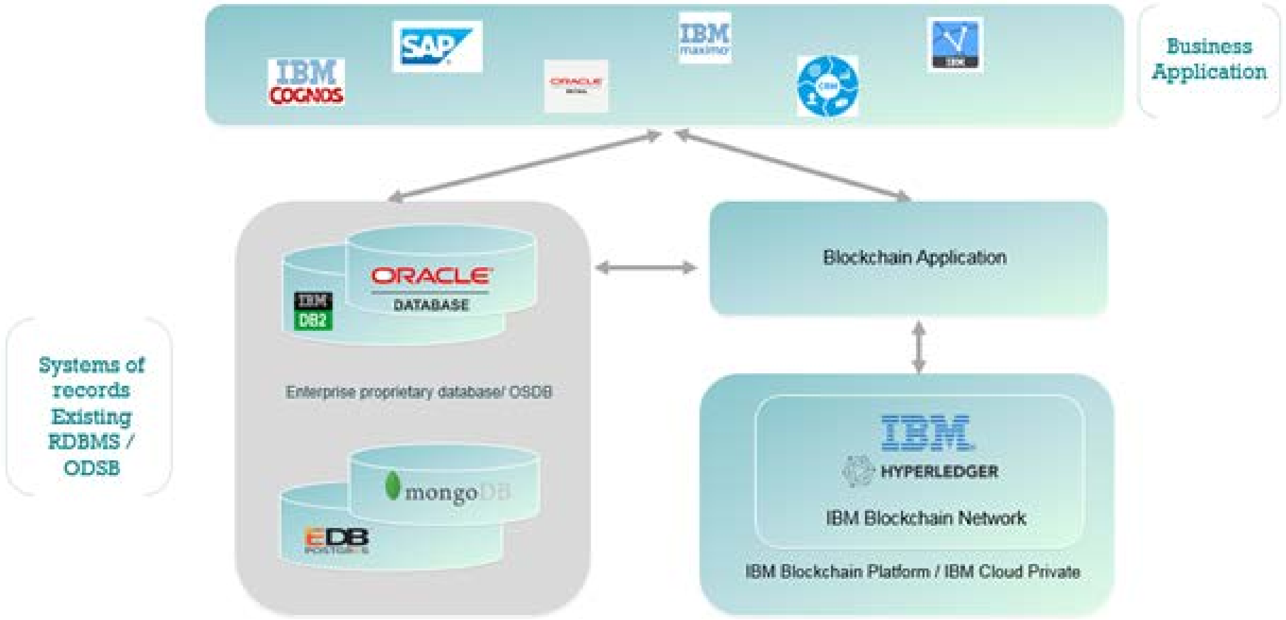


Figure 1. Architecture diagram

The architecture diagram in Figure 1 depicts the integration of business applications with an enterprise database such as Oracle Database, interacting with IBM Blockchain. The blockchain application (for example, Hyperledger Composer REST Server) help in communicating with the underlying Hyperledger Fabric and thus data flow happens between the blockchain application, enterprise databases, and Hyperledger Fabric.

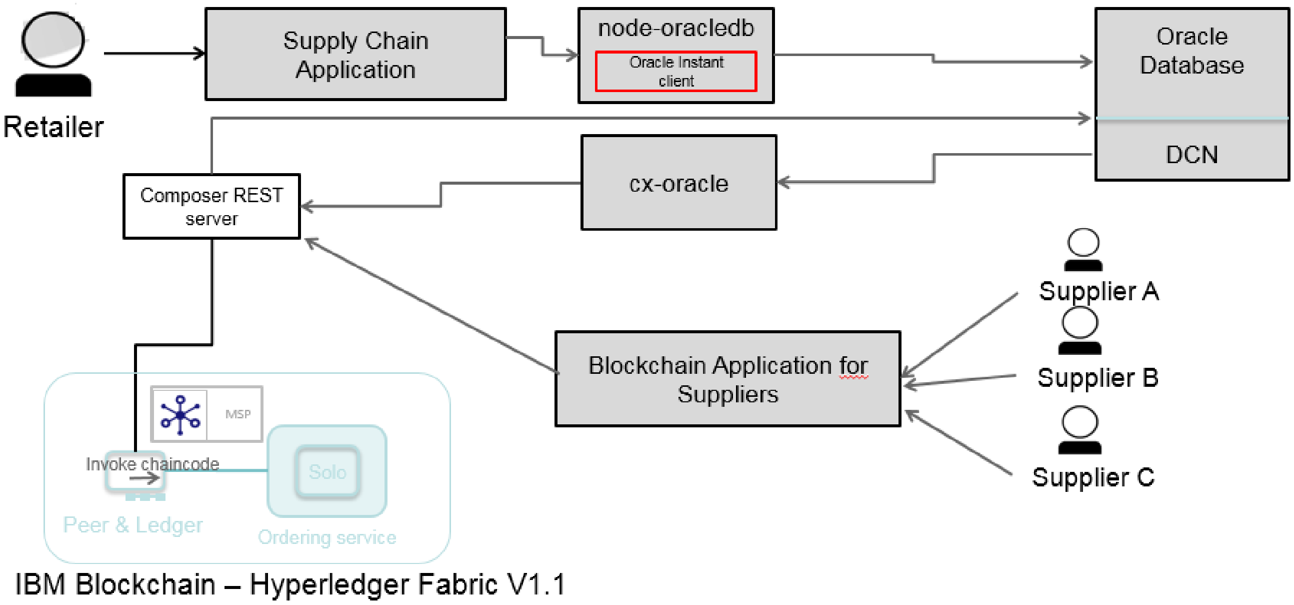


Figure 2. Dataflow diagram

Let's look at the various components and modules involved in this exercise.

- *cx_Oracle* is a Python extension module that can be used to enable access to Oracle Database. The recent version of *cx_Oracle* is 6 and has been tested to work with Python versions 2.7 and 3.4. *cx_Oracle* can be used with Oracle Database 11.2, 12.1, and 12.2 libraries.

- *node-oracledb* is an add-on for Node.js that allows high performance Oracle Database applications that are developed using Node.js and Python. The *node-oracledb* driver connects to an Oracle Database for fast and functional applications.

- Peers are a fundamental element of the network because they host ledgers and smart contracts.

- Orderer form ordering service is a communication medium that assures delivery of messages.

- Fabric CA is a certificate authority (CA) for Hyperledger Fabric. It allows registration of identities, issuance of enrollment certificates (ECerts), certificate renewal, and revocation.

- Smart contract or Chaincode helps in encapsulating the business logic. In our case, this represents the rules and regulations on which retailers and suppliers would interact.

- Composer is an application development framework that simplifies and expedites the creation of Hyperledger Fabric blockchain applications.

Retailer can interact with the enterprise application as a supply chain application for managing purchase orders, stores, inventory, and warehouses. For this demo, we considered order management system, which is part of the supply chain application. Node *oracledb* modules provide connectivity from the node application to the Oracle Database. After an entry is made in the Oracle Database, a database change notification will trigger an event which is captured by cx-oracle. cx-oracle will publish the event to the Composer REST Server which allows interaction with Hyperledger Fabric through Composer APIs.

Suppliers can interact and accept or reject the purchase order from a supplier using the blockchain application.

After the suppliers enter the quantity and submit a transaction, the respective transaction is validated against the smart contract and the result is captured through *cx_oracle* from Composer REST Server back to Oracle Database.

Steps

This section describes the steps to install Oracle Database and Hyperledger Fabric and thus setup supplier retailer application.

Install and configure Oracle Database and create the schema

1. Install Oracle Database 12.2 on AIX 7.2 as per the instructions provided at the following URL:

   https://docs.oracle.com/en/database/oracle/oracle-database/12.2/axdbi/index.html

2. After the Oracle Database is up and running, checkout the Oracle Database scripts at the following URL to create schema:

   https://github.com/harsha544/supplier-retailer-app/tree/master/oracle_tables

3. After setting up the Oracle environment using the `PATH`, `ORACLE_HOME`, and `ORACLE_SID` environment variables, invoke SQL Plus using the `sqlplus` command.

   ```
   $ sqlplus user/user@dbname
   ```

   Execute the wrapper.sql command to create the Oracle Database objects

   ```
   $@wrapper.sql
   ```

Install and configure node-oracledb and cx_Oracle

You need to install *node-oracledb* and *cx_Oracle* for Node.js and Python applications respectively running on Linux on Power Systems to access the Oracle Database. Refer to the following URL for detailed instructions:

https://developer.ibm.com/tutorials/connecting-python-nodejs-apps-oracle-db/

Install and configure Hyperledger Fabric

This section provides the steps to install and configure Hyperledger Fabric on Ubuntu 16.04 LTS ppc64le.

To install nvm, run the following command:

- Using cURL:

```
$ curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.33.11/install.sh | bash
```

- Using Wget:

```
$ wget -qO- https://raw.githubusercontent.com/creationix/nvm/v0.33.11/install.sh | bash
```

To install Python 2.X on Ubuntu, run the following command:

```
$ apt install python2.7
```

Note: Ensure you have following version of Docker and Docker Compose:

- Docker Engine 18.03 or later
- docker-compose 1.11.0 or later

Bootstrap the blockchain network

Refer to the steps at the following URL to bootstrap the blockchain network:

https://github.com/harsha544/supplier-retailer-app#bootstrapping-the-environment

Install and configure UI services

In this section you will install dependent npm modules and start UI services through the Node.js server that invokes the retailer as well as the supplier UI through a set of REST services.

1. Start the retailer UI.

   ```
   $  cd $HOME/supplier-retailer-app/UI/Retailer_UI
   $ npm install
   $ nohup node server.js 2>&1 &
   ```

   On successful completion, you can access the retailer UI at *http://IPADDR:8082/*

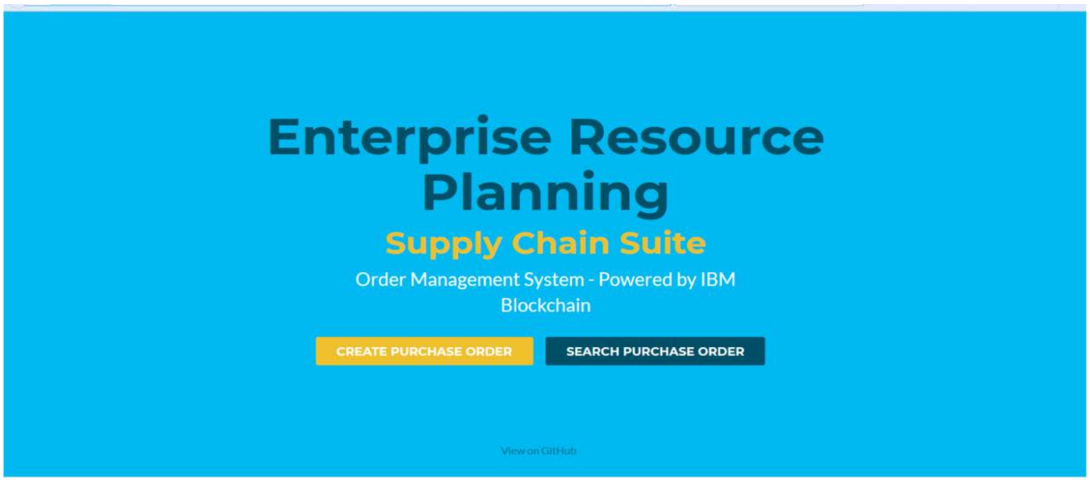

Enterprise Resource Planning
Supply Chain Suite
Order Management System - Powered by IBM Blockchain
CREATE PURCHASE ORDER
SEARCH PURCHASE ORDER
View on GitHub

Figure 3. Home page

2. Start the supplier UI

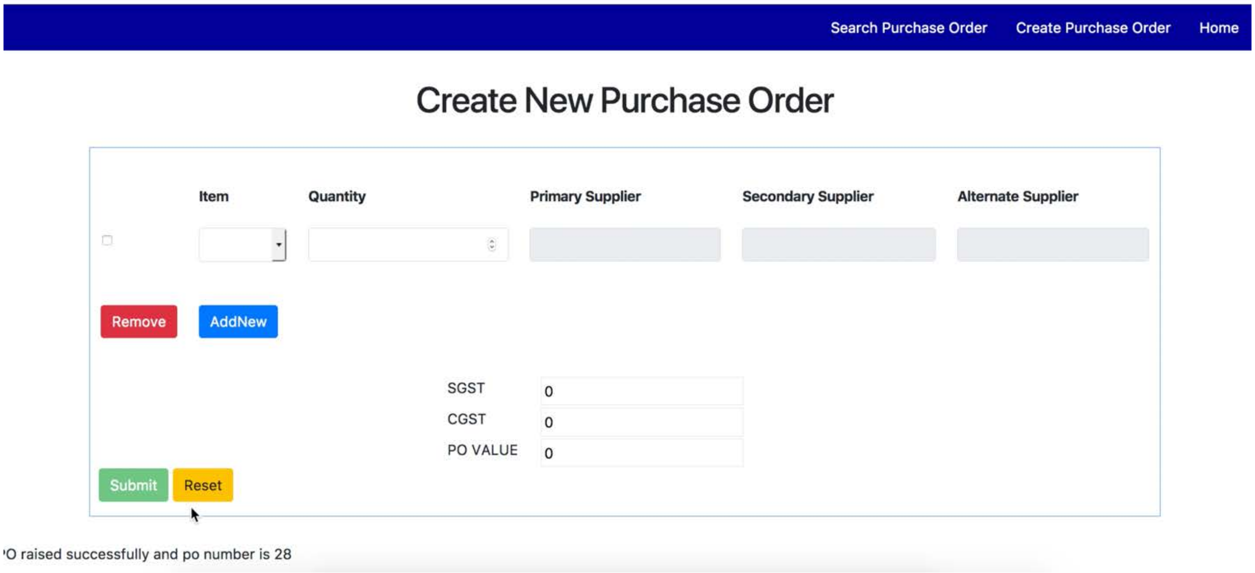
```
$ cd $HOME/supplier-retailer-app/UI/Supplier_UI
$ npm install
$ nohup node server.js 2>&1 &
```

On successful completion, you can access the supplier UI at *http://IPADDR:8083/*

Create a purchase order

In section we will see how to raise a new purchase order in the system. (the PO number will be autogenerated). Users can choose from pre-populated item lists and click **Add** to add items to the purchase order. Users must enter the item quantity against each item in the form. Based on the quantity and price of each item State Goods and Services Tax (SGST) and Central Goods and Services Tax (CGST) along with total value gets auto populated in the PO value of the page.

Upon submission, the PO along with the item details such as *item ID* and *quantit*y will be sent to the Blockchain network. The suppliers in the blockchain network will now bid the quantity against each item which they could supply and send their response back to the application. If all of the items could be supplied by the suppliers in the blockchain network, the status of the purchase order will be **Approved**. If all the items in the request could not be supplied by the suppliers in the blockchain network, then the status of the purchase order will be **Rejected** along with the comments from the blockchain network on how the modification must be done so that the purchase order gets approved.



Create New Purchase Order

□	Item	Quantity	Primary Supplier	Secondary Supplier	Alternate Supplier

SGST 0

CGST 0

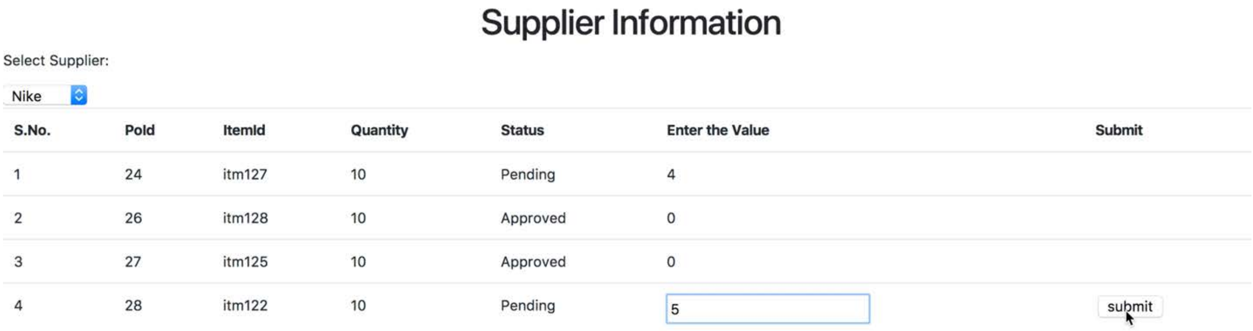
PO VALUE 0

'O raised successfully and po number is 28

Figure 4. Creating a new purchase order

Access supplier page

After generating the PO successfully, suppliers will now bid the quantity that they can supply against each item in the PO. To do so, access the supplier UI at https://IPADDR:8083


Supplier Information

Select Supplier:

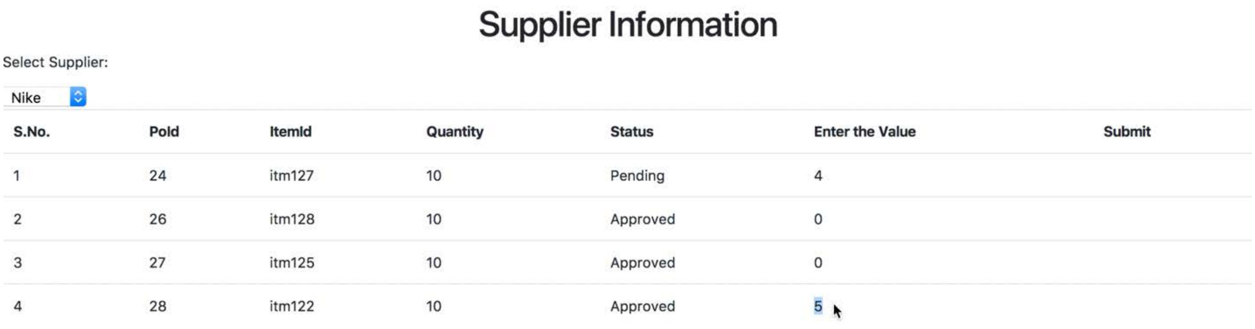
Nike

S.No.	PoId	ItemId	Quantity	Status	Enter the Value	Submit
1	24	itm127	10	Pending	4	
2	26	itm128	10	Approved	0	
3	27	itm125	10	Approved	0	
4	28	itm122	10	Pending	5	submit

Figure 5. Supplier home page

You can switch between the suppliers (for example, Nike/Adidas/Reebok) who are the participants of this blockchain network and specify the quantities to be supplied and click **submit**.

After recording the response from each supplier over the blockchain network, the PO gets evaluated and the quantity to be supplied for each supplier is shown (refer to Figure 6).



Supplier Information

Select Supplier:

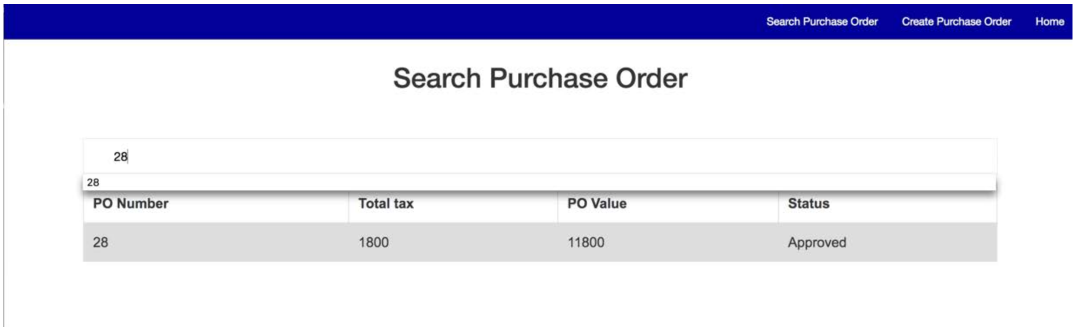
Nike

S.No.	PoId	ItemId	Quantity	Status	Enter the Value	Submit
1	24	itm127	10	Pending	4	
2	26	itm128	10	Approved	0	
3	27	itm125	10	Approved	0	
4	28	itm122	10	Approved	5	

Figure 6. Approved quantities

Search for the PO

The Search Purchase Order page is used to search an existing PO. User can enter the PO number and click **search**. The PO will be displayed on the page along with the **Total tax**, **PO Value** and **Status** fields of the PO.



Search Purchase Order

28

28

PO Number	Total tax	PO Value	Status
28	1800	11800	Approved

Figure 7: Searching a purchase order

Summary

The application discussed in this paper can be used as a reference to identify, implement, and integrate a functionality within an enterprise software with IBM Blockchain.

You can use the *cx_oracle* and *node-oracledb* modules to integrate Node.js and Python modules with Oracle Database. This application demonstrates the use of IBM Blockchain not only in the banking industry but in other sectors and domains as well.

Resources

- Why Linux on Power?
  https://www.ibm.com/developerworks/library/l-linux-on-power/index.html

- node-oracledb 2.3 documentation
  http://oracle.github.io/node-oracledb/doc/api.html

- cx_Oracle documentation
  http://cx-oracle.readthedocs.io/en/latest/

- IBM Power Systems
  https://www.ibm.com/it-infrastructure/power

- Hyperledger Fabric
  http://hyperledger-fabric.readthedocs.io/

- Installation of Oracle
  https://docs.oracle.com/en/database/oracle/oracle-database/12.2/axdbi/index.html

About the authors

Arun Sar

Arun Sar is a Senior Technical Staff Member in ISV organization of IBM Cognitive Systems business unit. As a open source database technology manager, he owns the responsibilities of building database ecosystem on IBM Power Systems. He can be reached at asar@us.ibm.com.

Krishna Harsha Voora

Krishna Harsha Voora has 6 years of experience with IBM as an administrator, a developer, and a consultant in UNIX® and Linux® operating systems. He is actively contributing to Hyperledger Fabric on Power and he evangelizes IBM Blockchain technology in various forums, conferences, and meet-ups. He contributes to Linux on Power ecosystem by porting the required software stack. He actively works on Linux on Power ISV Engagement. He can be reached at krishvoor@in.ibm.com.

Mithun Hallikerehundi

Mithun has 7 years of IT experience with Master of technology degree in computer science from Visveswaraya technological University and has been with IBM for the past 3 years as a technical consultant. He has experience in retail, aerospace, and defense domains. He is currently working as an Oracle ISV consultant performing Oracle product certifications on IBM AIX. He has good experience in Java™ and is also involved in helping ISVs to port their solutions to Linux on Power. He is interested in open source databases and Power AI. Mithun can be reached at mithunhr@in.ibm.com.

Srirama Sharma

Srirama Sharma has an overall IT experience of more than 14 years and has been with IBM for the last 7 years as a Technical Consultant leading ISV and open source ecosystem enablement at IBM Systems Development Lab (ISDL). He is the technical lead for the *Blockchain on Power* mission globally working closely with the Hyperledger Fabric community and evangelizes Blockchain technology in various forums, conferences, and meet-ups. He is also involved in building and enabling open source-based solutions supporting ISV and customer opportunities across different technologies such as private cloud and cognitive. He is also a key player in the financial software and systems industry. Srirama can be reached at sriramsh@in.ibm.com.

Yash Agrawal

Yash Agrawal has an overall IT experience of 6 months with a Bachelor of Technology degree in Computer Science from the PES University and is currently working as a cloud support associate (Linux) at Amazon Web Services. He has interned at IBM ISDL from Jan to July 2018 in ISV team. He helped contributors solve issues in Hyperledger Fabric. He has worked with open source technologies such as ONOS and software-defined networks. He is primarily interested in distributed community and open source technologies. Yash can be reached at yasagra1@in.ibm.com.

Thasmiya Noorain A N

Thasmiya has a Bachelor of Engineering degree in Information Science. Currently she is working at Infinera India as a software developer for Optical Line Systems. As an intern at IBM ISDL, she worked on a couple of Hyperledger Fabric projects. She is primarily interested in artificial intelligence, machine learning, and cloud computing technologies. Thasmiya can be reached at an.tasmiya@gmail.com.

Trademarks and special notices

References in this document to IBM products or services do not imply that IBM intends to make them available in every country.

IBM, the IBM logo, ibm.com, AIX, Power, Power Systems and POWER9 are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

The information provided in this document is distributed “AS IS” without any warranty, either express or implied.

The information in this document may include technical inaccuracies or typographical errors.

All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics may vary by customer.

Information concerning non-IBM products was obtained from a supplier of these products, published announcement material, or other publicly available sources and does not constitute an endorsement of such products by IBM. Sources for non-IBM list prices and performance numbers are taken from publicly available information, including vendor announcements and vendor worldwide homepages. IBM has not tested these products and cannot confirm the accuracy of performance, capability, or any other claims related to non-IBM products. Questions on the capability of non-IBM products should be addressed to the supplier of those products.

All statements regarding IBM future direction and intent are subject to change or withdrawal without notice and represent goals and objectives only. Contact your local IBM office or IBM authorized reseller for the full text of the specific Statement of Direction.

Some information addresses anticipated future capabilities. Such information is not intended as a definitive statement of a commitment to specific levels of performance, function or delivery schedules with respect to any future products. Such commitments are only made in IBM product announcements. The information is presented here to communicate IBM's current investment and development activities as a good faith effort to help with our customers' future planning.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

Photographs shown are of engineering prototypes. Changes may be incorporated in production models.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.