



GLOBAL FINANCE WITH BLOCKCHAIN

Develop with IBM Blockchain Platform extension for Visual
Studio code

Raheel Zubairy



Agenda

- Global Finance use case
- Using IBM Blockchain Platform extension for Visual Studio code
- Demo Global Finance with Blockchain App
- Creating a Node.js web application

Global Finance use case

- Inspired by Redbook Tutorial by Bob Dill
 - *Provides a walk through of blockchain*
 - *Provides a detailed background on the global finance use case*
 - *Creates a network through Hyperledger Composer*
 - *Implements a web app with user experience for the five participants*

Global Finance use case

- Five Participants
 - *Buyers*
 - *Sellers*
 - *Providers*
 - *Shippers*
 - *Finance Company*
- Provide traceability and transparency for **orders**
- Provides for dispute resolution

The basic stories

- As a Finance Organization, I want to see the finance related status of every order executed by my clients when they are using my credit services instantly and in real-time
 - *This will allow me to manage dispute resolution over the phone immediately rather than taking multiple weeks to resolve a dispute.*
- As a seller, I want to see the order, shipping and finance status of every sale in the system.
- As a buyer, I want to see the real-time status of every order.
- As a buyer, I want to be able to initiate a dispute with the click of a single button and provide all required data automatically to my finance organization
- As a manufacturer, I want to be able to see all open orders and the shipment status on all orders.
- As a shipper, I want to be able to interact with this system with as little change as possible on my end.

Participant Actions

■ Buyers can

- *create order providing*
 - the items
 - total amount
 - seller to purchase from
 - the finance company
- *purchase the order from seller*
- *cancel the order*
- *authorize payment to seller once items are delivered and verified*
- *dispute the order if not satisfied with the items received*

Participant Actions

■ Sellers can

- *order the items from a provider*
- *request payment from the buyer*
- *resolve a dispute*
- *refund if needed*

■ Providers can

- *request shipping from a shipper*
- *backorder on a order*
- *resolve a dispute*
- *refund if needed*

Participant Actions

- Shippers can
 - *provide delivering status as items are delivered*
 - *update the order once delivered*
 - *resolve a dispute*
 - *refund if needed*
- Finance Company can
 - *pay the seller once authorized from the buyer*

Transactions on Contract

- Buyers can
 - *CreateOrder, Buy, OrderCancel, AuthorizePayment, Dispute*
- Sellers can
 - *OrderFromSupplier, RequestPayment*
- Providers can
 - *RequestShipping, BackOrder*
- Shippers can
 - *Delivering, Delivered*
- Finance Company can
 - *Pay*
- Sellers, Providers, Shippers
 - *Resolve, Refund*

Order Tracking

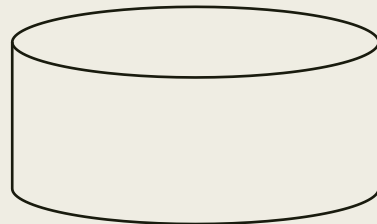
- An order has a status of:

- *Created*
- *Purchased*
- *Ordered from Provider*
- *Shipping Requested*
- *Delivering*
- *Delivered*
- *Payment Requested*
- *Authorize Payment*
- *Payment Processed*
- *Cancelled*
- *Backordered*
- *Dispute*
- *Dispute Resolved*
- *Refunded*

Blockchain Application Developer

■ World State

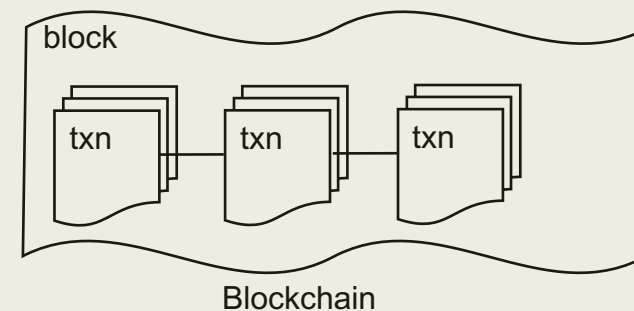
- *An ordinary database (e.g. key/value store)*
- *Stores the combined outputs of all transactions*



World state

■ Ledger

- *A linked list of blocks*
- *Each block describes a set of transactions*
- *Immutable – blocks cannot be tampered*



IBM Blockchain Platform extension for Visual Studio code

- Create contract
- Package contract
- Setup local fabric network
- Install and instantiate contract on the network

DEMO

Creating a Node.js web application

- Submit transactions
- Query transactions
- Create the participant experiences
- Provide admin console
- Display Blockchain to track all actions