User Study Documentation for Blockchain Enabler Platform

This user study documentation is provided for better understanding of the project and works as guide to navigate and work around the project.

Github link for the project: https://github.com/DataManagementLab/thesis-blockchainManager

Follow the readme for the project to install the dependencies and project in your system.
After the installation has been successfully done, go through these tasks in order to run and evaluate the project.

Task1: Setting up 2 Organization network locally

For creating a network with 2 orgs, Organisation needs to be invited by other organisation(owning the network) in order to join its network

To setup follow the instructions below.

- Initialise and create two Orgs CompanyA and CompanyB
- Add CompanyB to the network created by CompanyA
- CompanyB joins the companyA network

Clone the project and navigate to the folder location thesis/thesis-blockchainManager

Local Machine	**Local Machine**
go run main.go init CompanyA --local go run main.go create -u CompanyA	
	go run main.go init CompanyB --local go run main.go create -u CompanyB

go run main.go add -u CompanyA -z ~/.enabler/platform/CompanyB/CompanyB_network1/enabler/CompanyBOrg1_Invite.zip	
	go run main.go join -u CompanyB -z ~/.enabler/platform/CompanyA/CompanyA_network1/enabler/CompanyAOrg1_accept_transfer.zip
docker exec -it peer0.companyaorg1.example.com peer channel list	docker exec -it peer0.companyborg1.example.com peer channel list

Another mechanism to test is to check the height using these commands and match so that both are at same ledger height.

```
docker exec -it peer0.companyaorg1.example.com peer channel getinfo -c channelcompanyaorg1
```

```
docker exec -it peer0.companyborg1.example.com peer channel getinfo -c channelcompanyaorg1
```

Task 2 : Setting up 3 Organization network locally

- Initialize and create another organization CompanyC
- Add CompanyC to the network created by CompanyA
- CompanyC joins the network

NOTE: The last Organization to endorse needs to update it to the network using —update flag in sign.

```
go run main.go init CompanyC --local
```

```
go run main.go create -u CompanyC
go run main.go add -u CompanyA -z ~/.enabler/platform/CompanyC/CompanyC_network1/enabler/CompanyCOrg1_Invite.zip
```

```
go run main.go sign -u CompanyB --update -z ~/.enabler/platform/CompanyA/CompanyA_network1/enabler/CompanyCOrg1_sign_transfer.zip
```

```
go run main.go join -u CompanyC -z ~/.enabler/platform/CompanyA/CompanyA_network1/enabler/CompanyAOrg1_accept_transfer.zip
```

In order to test

```
docker exec -it peer0.companyaorg1.example.com peer channel list
```

```
docker exec -it peer0.companyborg1.example.com peer channel list
```

```
docker exec -it peer0.companycorg1.example.com peer channel list
```

Another mechanism to test is to check the height using these commands and match so that both are at same ledger height.

```
docker exec -it peer0.companyaorg1.example.com peer channel getinfo -c channelcompanyaorg1
```

```
docker exec -it peer0.companyborg1.example.com peer channel getinfo -c channelcompanyaorg1
```

```
docker exec -it peer0.companycorg1.example.com peer channel getinfo -c channelcompanyaorg1
```

Task 3: Self setup

- To the above 3 Organization setup add another organisation CompanyD.
- CompanyD sends the invite file to CompanyA and CompanyA adds CompanyD to its network.
- Sign the transfer file by CompanyB and CompanyC and then update it on to the network.
- Next join the network by CompanyD

To test run

```
docker exec -it peer0.companydorg1.example.com peer channel list
```

Another mechanism to test is to check the height using these commands and match so that both are at same ledger height.

```
docker exec -it peer0.companyaorg1.example.com peer channel getinfo -c
channelcompanyaorg1

docker exec -it peer0.companyborg1.example.com peer channel getinfo -c
channelcompanyaorg1

docker exec -it peer0.companycorg1.example.com peer channel getinfo -c
channelcompanyaorg1

docker exec -it peer0.companydorg1.example.com peer channel getinfo -c
channelcompanyaorg1
```

Task 4 : Setting up 2 Organization network Remotely

- Initialise and create organisations CompanyA and CompanyB on these different machines which are part of the cluster.
- CompanyB sends the invite file to the CompanyA and then CompanyA adds it to its network.
- CompanyA sends the accept_transfer file which contains the network configuration back to the CompanyB and then CompanyB uses this file to join the network created by CompanyA

This is a guide for the remote setup between different Host machines.
Follow the instructions below to create and join a network remotely.

Host Machine 1 **(User1- Kinshuk)**	**Host Machine 2** **(User2 Anonymous)**
go run main.go init CompanyA go run main.go create -u CompanyA	
	go run main.go init CompanyB go run main.go create -u CompanyB

	scp /home/kkislay/.enabler/platform/CompanyB/CompanyB_network1/enabler/CompanyBOrg1_Invite.zip kkislay@10.0.5.4:/home/kkislay/.enabler/platform/CompanyA/CompanyA_network1/enabler/CompanyBOrg1_Invite.zip
go run main.go add -u CompanyA -z ~/.enabler/platform/CompanyA/CompanyA_network1/enabler/CompanyBOrg1_Invite.zip	
scp /home/kkislay/.enabler/platform/CompanyA/CompanyA_network1/enabler/CompanyAOrg1_accept_transfer.zip kkislay@10.0.5.5:/home/kkislay/.enabler/platform/CompanyB/CompanyB_network1/enabler/CompanyAOrg1_accept_transfer.zip	
	go run main.go join -u CompanyB -z ~/.enabler/platform/CompanyB/CompanyB_network1/enabler/CompanyAOrg1_accept_transfer.zip
docker exec -it peer0.companyaorg1.example.com peer channel list	docker exec -it peer0.companyborg1.example.com peer channel list
docker exec -it peer0.companyaorg1.example.com peer channel getinfo -c channelcompanyaorg1	docker exec -it peer0.companyborg1.example.com peer channel getinfo -c channelcompanyaorg1

Kindly refer to this survey once you are done with each of the tasks.

https://s.surveyplanet.com/hfw8mrwt