



PRACTICAL 1

Aim: Install and understand Docker container, Node.js, Java and Hyperledger Fabric, Ethereum and perform necessary software installation on local machine/create instance on Cloud to run.

Background Information:

Docker is an open-source containerization platform used for developing, deploying, and managing applications in lightweight virtualized environments called containers. It is mainly used as a software development platform for developing distributed applications that work efficiently in different environments. By making the software system agnostic, developers don't have to worry about compatibility issues. Packaging apps into isolated environments (containers) also makes it easier to develop, deploy, maintain, and use applications.

Installation steps:

1. Download Docker: Go to the Docker website and download the appropriate Docker installer for your operating system. Or download docker from here, <https://www.docker.com/products/docker-desktop/>
2. Install Docker: Follow the on-screen instructions to install Docker
3. Verify installation: Open a terminal window and run the following command:

Command 1: #docker version

```
C:\Users\Pratik> docker version
Client:
  Cloud integration: v1.0.35+desktop.5
  Version:           24.0.6
  API version:       1.43
  Go version:        go1.20.7
  Git commit:        ed223bc
  Built:             Mon Sep  4 12:32:48 2023
  OS/Arch:           windows/amd64
  Context:            default

Server: Docker Desktop 4.25.0 (126437)
  Engine:
    Version:          24.0.6
    API version:      1.43 (minimum version 1.12)
    Go version:       go1.20.7
    Git commit:       la79695
    Built:            Mon Sep  4 12:32:16 2023
    OS/Arch:          linux/amd64
    Experimental:     false
  containerd:
    Version:          1.6.22
    GitCommit:        8165feabfdfe38c65b599c4993d227328c231fc
  runc:
    Version:          1.1.8
    GitCommit:        v1.1.8-0-g82f18fe
  docker-init:
    Version:          0.19.0
    GitCommit:        de40ad0

C:\Users\Pratik>
```

- 4 Start the Docker daemon: The Docker daemon is the service that runs Docker containers.
 - Test Docker: To test Docker, run the following command:
 - 5



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan

Command 2: # docker run hello-world

```
C:\Users\Pratik> docker run hello-world
Unable to find image 'hello-world:latest' locally
Latest: Pulling from library/hello-world
719385e32844: Pull complete
Digest: sha256:88ec0acaa3ec199d3b7eaf73588f4518c25f9d34f58ce9a0df68429c5af48e8d
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/
```

6. Pull a UBUNTU image from docker hub and run the CentOS container.
Make sure the Docker daemon is running.
7. Pull a UBUNTU image from docker hub and run

Command 3:

```
#docker pull ubuntu
```

```
#docker run -it ubuntu
```

8. Now Install and understand Hyperledger Fabric and Ethereum.

Command 4:

```
apt-get update && apt-get install -y \
soware-properes-common \
curl \
wget \
unzip \
git \
apt-uls \
build-essenal \
gcc \
g++ \
make \
curl -sL hps://deb.nodesource.com/setup_20.x | bash -
apt-get install -y nodejs
apt-get install -y openjdk-11-jdk
add-apt-repository ppa:ethereum/Ethereum
```



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan

```
apt-get update -y  
apt-get install -y solc  
wget https://go.dev/dl/go1.21.3.linux-amd64.tar.gz  
tar -C /usr/local -xzf go1.21.3.linux-amd64.tar.gz  
rm go1.21.3.linux-amd64.tar.gz
```

```
apt-get install -y \
```

```
docker.io \
```

```
libtool \
```

```
libltdl-dev \
```

```
libssl-dev \
```

```
autoconf \
```

```
automake \
```

```
bison \
```

```
libboost-all-dev \
```

```
libgflags-dev \
```

```
libprotobuf-dev \
```

```
libleveldb-dev \
```

```
libsnapy-dev \
```

```
libsodium-dev \
```

```
liblz4-tool
```

```
service --status-all
```



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan

PRACTICAL 2

Aim: Create and deploy a block chain network using Hyperledger Fabric SDK for Java.

Background Information:

Hyperledger Fabric:

Hyperledger Fabric is a permissioned blockchain platform that offers a modular architecture, allowing components such as consensus and membership services to be plug-and-play. It is designed for use in enterprise contexts and supports smart contracts written in various programming languages.

SDK for Java:

The Hyperledger Fabric SDK for Java enables Java applications to interact with Hyperledger Fabric networks. It provides a set of APIs for creating, deploying, and interacting with smart contracts, as well as managing identities and transactions.

Command 1:

`Docker –version`

Command 2:

`docker –version`

Command 3:

`docker-compose –version`

Command 4:

`export FABRIC_HOME=/path/to/fabric-sdk-java`

Command 5:

`export JAVA_HOME=/path/to/jdk11.0.17`

Command 6:

`git clone https://github.com/hyperledger/fabric-samples.git`

Command 7:

`mkdir pract2`

`cd pract2`

`ls`



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan

Command 8:

```
curl -sSL hps://bit.ly/2ysbOFE | bash -s
```

Command 9:

```
cd fabric-samples/test-network
```

Command 10:

```
./network.sh up
```

```
unknown flag: --v
alex@Alex:~$ docker-compose --version
Docker Compose version v2.23.0-desktop.1
alex@Alex:~$ export FABRIC_HOME=/path/to/fabric-sdk-java
alex@Alex:~$ export JAVA_HOME=/path/to/jdk11.0.17
alex@Alex:~$ git clone https://github.com/hyperledger/fabric-samples.git
Cloning into 'fabric-samples'...
remote: Enumerating objects: 13012, done.
remote: Counting objects: 100% (1044/1044), done.
remote: Compressing objects: 100% (564/564), done.
remote: Total 13012 (delta 541), reused 757 (delta 429), pack-reused 11968
Receiving objects: 100% (13012/13012), 23.00 MiB | 2.27 MiB/s, done.
Resolving deltas: 100% (6995/6995), done.
alex@Alex:~$ mkdir p2
alex@Alex:~$ cd p2
alex@Alex:~/p2$ curl -sSL https://bit.ly/2ysbOFE | bash -s

Clone hyperledger/fabric-samples repo
==> Cloning hyperledger/fabric-samples repo
Cloning into 'fabric-samples'...
remote: Enumerating objects: 13012, done.
remote: Counting objects: 100% (1044/1044), done.
remote: Compressing objects: 100% (564/564), done.
remote: Total 13012 (delta 541), reused 757 (delta 429), pack-reused 11968
Receiving objects: 100% (13012/13012), 23.00 MiB | 2.27 MiB/s, done.
Resolving deltas: 100% (6995/6995), done.
fabric-samples v2.5.4 does not exist, defaulting to main. fabric-samples main branch is intended to work with recent versions of fabric.

Pull Hyperledger Fabric binaries
==> Downloading version 2.5.4 platform specific fabric binaries
==> Downloading: https://github.com/hyperledger/fabric/releases/download/v2.5.4/hyperledger-fabric-linux-amd64-2.5.4.tar.gz
  % Total    % Received % Xferd  Average Speed   Time   Time     Current
          Dload  Upload Total Spent   Left Speed
  0      0      0      0      0      0      0      0      0      0
100  99.3M  100  99.3M      0      0:00:47  0:00:47      0      0
==> Done.
==> Downloading version 1.5.7 platform specific fabric-ca-client binary
==> Downloading: https://github.com/hyperledger/fabric-ca/releases/download/v1.5.7/hyperledger-fabric-ca-linux-amd64-1.5.7.tar.gz
```



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan

```
Pull Hyperledger Fabric docker images

FABRIC_IMAGES: peer orderer ccenv tools baseos
====> Pulling fabric Images
=====> docker.io/hyperledger/fabric-peer:2.5.4
2.5.4: Pulling from hyperledger/fabric-peer
01085d60b3a6: Pull complete
5920bb1ab585: Pull complete
0c13247db338: Pull complete
f1ebf2febfff: Pull complete
7ba246813ff2: Pull complete
20e090879749: Pull complete
Digest: sha256:c2e735a3cb8250c47ed3589294b3f0c078004ec001b949710f334736651e106d
Status: Downloaded newer image for hyperledger/fabric-peer:2.5.4
docker.io/hyperledger/fabric-peer:2.5.4

What's Next?
  View a summary of image vulnerabilities and recommendations + docker scout quickview docker.io/hyperledger/fabric-peer:2.5.4
=====> docker.io/hyperledger/fabric-orderer:2.5.4
2.5.4: Pulling from hyperledger/fabric-orderer
01085d60b3a6: Already exists
0ec39628d119: Pull complete
c9ef912f2449: Pull complete
5f267c8c968e: Pull complete
54a738441be6: Pull complete
d6c2bf9dde2f: Pull complete
Digest: sha256:1a7144705b435062f4be2886de98d0408165cf51326ec721c3f58b40bf8bf139
Status: Downloaded newer image for hyperledger/fabric-orderer:2.5.4
docker.io/hyperledger/fabric-orderer:2.5.4
```

```
alex@alex:~/p2$ ./network.sh up
Using docker and docker-compose
Starting nodes with CLI timeout of '5' tries and CLI delay of '3' seconds and using database 'leveldb' with crypto from 'cryptogen'
LOCAL_VERSION=v2.5.4
DOCKER_IMAGE_VERSION=v2.5.4
/home/alex/p2/fabric-samples/test-network/../bin/cryptogen
generating certificates using cryptogen tool
Creating Org1 Identities
+ cryptogen generate --config=../organizations/cryptogen/crypto-config-org1.yaml --output=organizations
org1.example.com
+ res=0
Creating Org2 Identities
```

Command 11:

./network.sh down



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan

```
Generating CCP Files for Org1 and Org2
[+] Building 0.0s (0/0)
[+] Running 4/4
  ● Container peer0.org2.example.com Started
  ● Container peer0.org1.example.com Started
  ● Container orderer.example.com Started
  ● Container cli Started
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
6416d86e3122 hyperledger/fabric-tools:latest "/bin/bash" 8 seconds ago Up 1 second docker:default 2.0s
05c327fd0eb5 hyperledger/fabric-peer:latest "peer node start" 10 seconds ago Up 5 seconds 0.0.0.0:7051->7051/tcp, 0.0.0.0:9444->9444/tcp 2.1s
c46b65c2a8fd hyperledger/fabric-orderer:latest "orderer" 10 seconds ago Up 4 seconds 0.0.0.0:7050->7050/tcp, 0.0.0.0:7053->7053/tcp, 0.0.0.0:9443->9443/tcp 1.6s
886b1f4edbd hyperledger/fabric-peer:latest "peer node start" 10 seconds ago Up 5 seconds 0.0.0.0:9051->9051/tcp, 7051/tcp, 0.0.0.0:9445->9445/tcp 0.9s
346ce966a5f 975aae09sc0b "peer node start" 2 days ago Created
0e90019ab0ff e090019ab0ff Created
efaf3e3f82c59 e0dcbfe2379 "bash" 3 days ago Exited (0) 3 days ago
7f3c740f175c e0dcbfe2379 "/bin/bash" 3 weeks ago Exited (255) 3 days ago
cf1817306892 e0dcbfe2379 "/bin/bash" 3 weeks ago Exited (0) 3 weeks ago
76a701609065 e0dcbfe2379 "/bin/bash" 3 weeks ago Exited (0) 3 weeks ago
alex@alex:~/p2/fabric-samples/test-network$ ./network.sh down
Using docker and docker-compose
Stopping network
[+] Running 12/12
  ● Container cli Removed
  ● Container orderer.example.com Removed 1.7s
  ● Container peer0.org2.example.com Removed 1.7s
  ● Container peer0.org1.example.com Removed 4.3s
  ● Volume compose_peer0.org1.example.com Removed 1.6s
  ● Volume compose_orderer2.example.com Removed 0.1s
  ● Volume compose_orderer3.example.com Removed 0.0s
  ● Volume compose_orderer4.example.com Removed 0.1s
  ● Volume compose_peer0.org2.example.com Removed 0.1s
  ● Volume compose_peer0.org3.example.com Removed 0.0s
  ● Volume compose_orderer.example.com Removed 0.1s
  ● Network fabric_test Removed 0.4s
Error response from daemon: get docker_orderer.example.com: no such volume
Error response from daemon: get docker_peer0.org1.example.com: no such volume
Error response from daemon: get docker_peer0.org2.example.com: no such volume
Removing remaining containers
Removing generated chaincode docker images
```

Command 12:

```
./network.sh up createChannel
```

```
alex@Alex:~/p2/fabric-samples/test-network$ ./network.sh up createChannel
Using docker and docker-compose
Creating channel 'mychannel'.
if network is not up, starting nodes with CLI timeout of '5' tries and CLI delay of '3' seconds and using database 'leveldb' with crypto from 'cryptogen'
Bringing up network
LOCAL_VERSION=v2.5.4
DOCKER_IMAGE_VERSION=v2.5.4
/home/alex/p2/fabric-samples/test-network/../bin/cryptogen
generating certificates using cryptogen tool
creating Org1 Identities
+ cryptogen generate --config=../organizations/cryptogen/crypto-config-org1.yaml --output=organizations
org1.example.com
+ res=0
Creating Org2 Identities
+ cryptogen generate --config=../organizations/cryptogen/crypto-config-org2.yaml --output=organizations
org2.example.com
+ res=0
treating Orderer Org Identities
+ cryptogen generate --config=../organizations/cryptogen/crypto-config-orderer.yaml --output=organizations
+ res=0
Generating CCP files for Org1 and Org2
[+] Building 0.0s (0/0)
[+] Running 8/8
  ● Network_fabric_test Created 0.1s
  ● Volume "compose_peer0.org1.example.com" Created 0.0s
  ● Volume "compose_peer0.org2.example.com" Created 0.0s
  ● Volume "compose_orderer.example.com" Created 0.0s
  ● Container orderer.example.com Started 0.2s
  ● Container peer0.org1.example.com Started 0.2s
  ● Container peer0.org2.example.com Started 0.2s
  ● Container cli Started 0.1s
```

Command 13:

```
cd asset-transfer-basic
```

Command 14:

```
cd chaincode-java
```

Command 15:



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan

docker build .



PRACTICAL 3

Aim: Interact with a block chain network. Execute transactions and requests against a block chain network by creating an app to test the network and its rules.

Background Information:

Creating a blockchain app involves understanding decentralized ledgers, smart contracts, and user wallets. Use Web3 libraries, test on blockchain testnets, prioritize security, and deploy on the live network after testing. Transactions require user initiation, consensus verification, and may involve fees.

Command 1:

```
git clone https://github.com/IBM/blockchain-application-using-fabric-java-sdk/
```

```
root@DESKTOP-CU5U5QE:~# git clone https://github.com/IBM/blockchain-application-using-fabric-java-sdk
Cloning into 'blockchain-application-using-fabric-java-sdk'...
remote: Enumerating objects: 810, done.
remote: Counting objects: 100% (99/99), done.
remote: Compressing objects: 100% (19/19), done.
remote: Total 810 (delta 86), reused 80 (delta 80), pack-reused 711
Receiving objects: 100% (810/810), 728.73 KiB | 2.38 MiB/s, done.
Resolving deltas: 100% (285/285), done.
```

Command 2:

```
cd /blockchain-application-using-fabric-java-sdk/
```

```
root@DESKTOP-CU5U5QE:~# cd blockchain-application-using-fabric-java-sdk
root@DESKTOP-CU5U5QE:~/blockchain-application-using-fabric-java-sdk# ll
total 68
drwxr-xr-x 7 root root 4096 Nov 30 13:36 .
drwx----- 8 root root 4096 Nov 30 13:36 ../
drwxr-xr-x 8 root root 4096 Nov 30 13:36 .git/
-rw-r--r-- 1 root root 569 Nov 30 13:36 CONTRIBUTING.md
-rw-r--r-- 1 root root 1549 Nov 30 13:36 DEBUGGING.md
-rw-r--r-- 1 root root 11357 Nov 30 13:36 LICENSE
-rw-r--r-- 1 root root 3131 Nov 30 13:36 MAINTAINERS.md
-rw-r--r-- 1 root root 15582 Nov 30 13:36 README.md
drwxr-xr-x 2 root root 4096 Nov 30 13:36 images/
drwxr-xr-x 3 root root 4096 Nov 30 13:36 java/
drwxr-xr-x 2 root root 4096 Nov 30 13:36 network/
drwxr-xr-x 5 root root 4096 Nov 30 13:36 network_resources/
root@DESKTOP-CU5U5QE:~/blockchain-application-using-fabric-java-sdk# cd network
root@DESKTOP-CU5U5QE:~/blockchain-application-using-fabric-java-sdk/network# chmod +x build.sh stop.sh teardown.sh
root@DESKTOP-CU5U5QE:~/blockchain-application-using-fabric-java-sdk/network# ./build.sh generate
```

Command 3:

```
cd network
```

```
chmod +x build.sh stop.sh teardown.sh
```



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan

```
root@DESKTOP-CU5U5QE:~/blockchain-application-using-fabric-java-sdk# cd network
root@DESKTOP-CU5U5QE:~/blockchain-application-using-fabric-java-sdk/network# chmod +x build.sh stop.sh teardown.sh
```

/build.sh generate – after generating if error show

If below error is shown then delete the container as shown in below ss.

```
✓ Network network_custom      Created      0.0s
✗ Container orderer.example.com Creating    0.0s
✗ Container ca_peerOrg1      Creating    0.0s
✗ Container ca_peerOrg2      Creating    0.0s
Error response from daemon: Conflict. The container name "/orderer.example.com" is already in use by container "db3d79657ebcd2c896ba84abdc91b02dd8366ec7b24ad5458693968b863d06". You have to remove (or rename) that container to be able to reuse that name.
root@DESKTOP-CU5U5QE:~/blockchain-application-using-fabric-java-sdk/network# ./network.sh up
-bash: ./network.sh: No such file or directory
```

Command 4:

/build.sh generate

```
root@DESKTOP-CU5U5QE:~/blockchain-application-using-fabric-java-sdk/network# ./build.sh generate

Stopping the previous network (if any)
[+] Running 3/3
  ✓ Container ca_peerOrg1   Removed
  ✓ Container ca_peerOrg2   Removed
  ✓ Network network_custom  Removed

Setting up the Hyperledger Fabric 1.1 network
[+] Building 0.0s (0/0)
[+] Running 8/8
  ✓ Network network_custom      Created
  ✓ Container orderer.example.com Started
  ✓ Container ca_peerOrg1      Started
  ✓ Container ca_peerOrg2      Started
  ✓ Container peer0.org2.example.com Started
  ✓ Container peer0.org1.example.com Started
  ✓ Container peer1.org2.example.com Started
  ✓ Container peer1.org1.example.com Started

Network setup completed!!
```

Command 5:

docker-compose -f docker-compose.yaml up

```
bash: ./network.sh: No such file or directory
root@DESKTOP-CU5U5QE:~/blockchain-application-using-fabric-java-sdk/network# docker-compose -f docker-compose.yaml up

Usage: docker compose [OPTIONS] COMMAND
Define and run multi-container applications with Docker.

Options:
  --ansi string            Control when to print ANSI control characters ("never"|"always"|"auto") (default "auto")
  --compatibility           Run compose in backward compatibility mode
  --dry-run                 Execute command in dry run mode
  --env-file stringArray   Specify an alternate environment file.
  -f, --file stringArray   Compose configuration files
  --parallel int            Control max parallelism, -1 for unlimited (default -1)
  --profile stringArray    Specify a profile to enable
  --progress string         Set type of progress output (auto, tty, plain, quiet) (default "auto")
  --project-directory string Specify an alternate working directory
                           (default: the path of the, first specified, Compose file)
  -p, --project-name string Project name

Commands:
```

Command 6:

cd .. /java/

```
root@DESKTOP-CU5U5QE:~/blockchain-application-using-fabric-java-sdk/network# cd ../java/
root@DESKTOP-CU5U5QE:~/blockchain-application-using-fabric-java-sdk/java# java -cp blockchain-clie
```

Command 7:

apt install maven



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan

Command 8:

mvn install

```
root@DESKTOP-CU5U5QE:~/blockchain-application-using-fabric-java-sdk/java# mvn install  
[INFO] Scanning for projects...  
[INFO]  
[INFO] -----< blockchain-java-sdk:blockchain-java-sdk >-----  
[INFO] Building blockchain-java-sdk 0.0.1-SNAPSHOT  
[INFO] -----[ jar ]-----
```

Command 9:

cd target/

```
mv blockchain-java-sdk-0.0.1-SNAPSHOT-jar-with-dependencies.jar blockchain-client.jar
```

```
cp blockchain-client.jar ../../network_resources
```

```
cd ../../network_resources
```

```
java -cp blockchain-client.jar org.example.network.CreateChannel
```

```
root@DESKTOP-CUSU5QE:~/blockchain-application-using-fabric-java-sdk/java# cd target/
root@DESKTOP-CUSU5QE:~/blockchain-application-using-fabric-java-sdk/java/target# mv blockchain-java-sdk-0.0.1-SNAPSHOT-jar-with-dependencies.jar blockchain-client.jar
root@DESKTOP-CUSU5QE:~/blockchain-application-using-fabric-java-sdk/java/target# cp blockchain-client.jar ../../network_resources
root@DESKTOP-CUSU5QE:~/blockchain-application-using-fabric-java-sdk/java/target# cd ../../network_resources
root@DESKTOP-CUSU5QE:~/blockchain-application-using-fabric-java-sdk/network_resources# java -cp blockchain-client.jar org.example.network.CreateChannel
[INFO] [WARN] No address could be found for listener (org.hyperledger.fabric.sdk.TLSServerConfig)
```

Command 10:

```
java -cp blockchain-client.jar org.example.user.RegisterEnrollUser
```

```
root@DESKTOP-CUSU5QE:~/blockchain-application-using-fabric-java-sdk/network_resources# java -cp blockchain-client.jar org.example.user.RegisterEnrollUser
Nov 30, 2023 2:24:00 PM org.example.util.Util deleteDirectory
INFO: Deleting - users
log4j:WARN No appenders could be found for logger (org.hyperledger.fabric.sdk.helper.Config).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/2.1/faq.html#noconfig for more info.
Nov 30, 2023 2:24:01 PM org.example.client.CAClient enrollAdminUser
INFO: CA -http://localhost:7054 Enrolled Admin.
Nov 30, 2023 2:24:01 PM org.example.client.CAClient registerUser
INFO: CA -http://localhost:7054 Registered User - user170133441172
Nov 30, 2023 2:24:01 PM org.example.client.CAClient enrollUser
INFO: CA -http://localhost:7054 Enrolled User - user170133441172
```



PRACTICAL 4

Aim:

Deploy an asset-transfer app using block chain. Learn app development within a Hyperledger Fabric network.

Background Information:

Hyperledger Fabric is a permissioned blockchain platform for enterprise. It features a modular architecture, smart contracts (chaincode), and supports private channels. Development involves network setup, chaincode coding, user registration, and application development. Security considerations include access control and encryption. Deploy the application after testing on a server or cloud platform.

Command 1:

```
$ cd fabric-samples/test-network
```

```
CONTRIBUTING.md      asset-transfer-ledger-queries      hardware-security-module      token-erc-20
LICENSE              asset-transfer-private-data      high-throughput                token-erc-721
MAINTAINERS.md       asset-transfer-sbe             off_chain_data                token-sdk
README.md            asset-transfer-secured-agreement test-application               token-utxo
SECURITY.md          auction-dutch                  test-network
alex@Alex:~/fabric-samples$ cd test-network
alex@Alex:~/fabric-samples/test-network$ ls
CHAINCODE_AS_A_SERVICE_TUTORIAL.md  bft-config  monitordocker.sh  organizations      setOrgEnv.sh
README.md                      compose    network.config   prometheus-grafana  system-genesis-block
addOrg3                         configtx   network.sh        scripts
alex@Alex:~/fabric-samples/test-network$ ll
total 100
drwxr-xr-x 10 alex alex 4096 Dec  2 20:22 .
drwxr-xr-x 27 alex alex 4096 Dec  2 20:22 ..
-rw-r--r--  1 alex alex  344 Dec  2 20:22 .gitignore
-rw-r--r--  1 alex alex 13717 Dec  2 20:22 CHAINCODE_AS_A_SERVICE_TUTORIAL.md
-rw-r--r--  1 alex alex  3565 Dec  2 20:22 README.md
drwxr-xr-x  4 alex alex 4096 Dec  2 20:22 addOrg3/
drwxr-xr-x  2 alex alex 4096 Dec  2 20:22 bft-config/
drwxr-xr-x  4 alex alex 4096 Dec  2 20:22 compose/
drwxr-xr-x  2 alex alex 4096 Dec  2 20:22 configtx/
-rw-r--r-x  1 alex alex  774 Dec  2 20:22 monitordocker.sh*
-rw-r--r--  1 alex alex 1482 Dec  2 20:22 network.config
-rw-r--r-x  1 alex alex 23046 Dec  2 20:22 network.sh*
drwxr-xr-x  5 alex alex 4096 Dec  2 20:22 organizations/
drwxr-xr-x  5 alex alex 4096 Dec  2 20:22 prometheus-grafana/
drwxr-xr-x  3 alex alex 4096 Dec  2 20:22 scripts/
-rw-r--r-x  1 alex alex 2291 Dec  2 20:22 setOrgEnv.sh*
drwxr-xr-x  2 alex alex 4096 Dec  2 20:22 system-genesis-block/
alex@Alex:~/fabric-samples/test-network$
```

Command 2:



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan

```
./network.sh down
```

```
$ ./network.sh up createChannel -c mychannel
```

Command 3:

```
$ ./network.sh deployCC -ccn secured -ccp ..asset-transfer-secured-agreement/chaincode-go/ -ccl go -ccep "OR('Org1MSP.peer','Org2MSP.peer')"
```

```
Chaincode definition committed on channel 'mychannel'
  using organization '
querying chaincode definition on peer0.org1 on channel 'mychannel'...
attempting to query committed status on peer0.org1, Retry after 3 seconds.
+ peer lifecycle chaincode querycommitted --channelID mychannel --name secured
- res=0
Committed chaincode definition for chaincode 'secured' on channel 'mychannel'.
Version: 1.0.1, Sequence: 1, Endorsement Plugin: escc, Validation Plugin: vscc, Approvals: [Org1MSP: true, Org2MSP: true]
Query chaincode definition successful on peer0.org1 on channel 'mychannel'
  using organization '
querying chaincode definition on peer0.org2 on channel 'mychannel'...
attempting to query committed status on peer0.org2, Retry after 3 seconds.
+ peer lifecycle chaincode querycommitted --channelID mychannel --name secured
- res=0
Committed chaincode definition for chaincode 'secured' on channel 'mychannel'.
Version: 1.0.1, Sequence: 1, Endorsement Plugin: escc, Validation Plugin: vscc, Approvals: [Org1MSP: true, Org2MSP: true]
Query chaincode definition successful on peer0.org2 on channel 'mychannel'
  chaincode initialization is not required
[1] ->/v1/rpc/test-network --
```

```
- CC_SRC_LANGUAGE: go
- CC_VERSION: 1.0.1
Rendering dependencies at ./asset-transfer-secured-agreement/chaincode-go/
~/hyperledger/prac2/fabric-samples/asset-transfer-secured-agreement/chaincode-go ~/hyperledger/prac2/fabric-samples/test-net
twerk
go: downloading github.com/hyperledger/fabric-contract-api-go v1.2.1
go: downloading github.com/hyperledger/fabric-chaincode-go v0.0.0-20230228194215-b84622bab7a
go: downloading google.golang.org/protobuf v1.28.1
go: downloading github.com/hyperledger/fabric-protos-go v0.3.0
go: downloading google.golang.org/grpc v1.53.0
go: downloading github.com/xipipuu/gojonschema v1.2.0
go: downloading github.com/go-openapi/spec v0.20.8
go: downloading github.com/gobuffalo/packr v1.30.1
go: downloading google.golang.org/genproto v0.0.0-20230110181048-76db0878b65f
go: downloading golang.org/x/net v0.7.0
go: downloading github.com/xipipuu/jsonreference v0.0.0-20180127040603-bd5ef7bd5415
go: downloading github.com/go-openapi/jsonpointer v0.19.5
go: downloading github.com/go-openapi/jsonreference v0.20.0
go: downloading github.com/go-openapi/swag v0.21.1
go: downloading golang.org/x/sys v0.5.0
go: downloading github.com/gobuffalo/envy v1.10.1
go: downloading github.com/gobuffalo/packd v1.0.1
go: downloading github.com/xipipuu/gojsonpointer v0.0.0-20190905194746-02993c407fb
go: downloading github.com/mailru/easyjson v0.7.7
go: downloading github.com/joho/godotenv v1.4.0
go: downloading github.com/roopepe/go-internal v1.8.1
go: downloading github.com/josharian/intern v1.0.0

[1] 8:bash* ./network.sh deployCC* 69:57 27-Nov-22
```



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan

Command 4:

Set the environment variables to operate as Org1 (posix compl shell)

```
$ export PATH=${PWD}/..bin:${PWD}:$PATH
$ export FABRIC_CFG_PATH=${PWD}/config/
$ export CORE_PEER_TLS_ENABLED=true
$ export CORE_PEER_LOCALMSPID="Org1MSP"
$ export
CORE_PEER_MSPCONFIGPATH=${PWD}/organizations/peerOrganizations/or
g1
.example.com/users/Admin@org1.example.com/msp
$ export
CORE_PEER_TLS_ROOTCERT_FILE=${PWD}/organizations/peerOrganizations
/o rg1.example.com/peers/peer0.org1.example.com/tls/ca.crt
$ export CORE_PEER_ADDRESS=localhost:7051
```



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan

Command 5:

Set the environment variables to operate as Org2

```
$ export PATH=${PWD}/..bin:${PWD}:$PATH  
$ export FABRIC_CFG_PATH=${PWD}/config/  
$ export CORE_PEER_TLS_ENABLED=true  
$ export CORE_PEER_LOCALMSPID="Org2MSP"  
$ export  
CORE_PEER_MSPCONFIGPATH=${PWD}/organizations/peerOrganizations/or  
g2  
.example.com/users/Admin@org2.example.com/msp  
$ export  
CORE_PEER_TLS_ROOTCERT_FILE=${PWD}/organizations/peerOrganizations  
/o rg2.example.com/peers/peer0.org2example.com/tls/ca.crt  
$ export CORE_PEER_ADDRESS=localhost:9051
```

Command 6:

Create an asset

Operate from the Org1 terminal (top Terminal):

```
$ export ASSET_PROPERTIES=$(echo -n  
{"object_type":"asset_properties","color":"blue","size":35,"salt":  
a94a8fe5ccb19ba61c4c0873d391e987982fbbd3}" | base64 | tr -d '\n')
```

```
subodh@subodh-HP-Laptop-15-bs0xx:~/hyperledger/prac2/fabric-samples/test-network$ peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls --cafile "${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n secured -c '{"function":"CreateAsset","Args":[{"A new asset for Org1MSP"}]} --transient "{\"asset_properties\":\"$ASSET_PROPERTIES\"}"  
2019-11-27 20:49:30.981 UTC [chaincodeEnd] chaincodeInvokeOrQuery [chaincode invoke successful. result: status:  
200 payload:"d9923f21b770adbc79cbcc47a3aeccc81dc7f030bd129155301ce393  
2 be7fbcc"
```

Command 7:

```
$ export  
ASSET_ID=d9923f21b770adbc79cbcc47a3aeccc81dc7f030bd129155301ce393  
2 be7fbcc
```

Success message:

```
{"object_type":"asset_properties","color":"blue","size":35,"salt": "a94a8fe5ccb  
19ba61c4c0873d391e987982fbbd3"}
```

```
subodh@subodh-HP-Laptop-15-bs0xx:~/hyperledger/prac2/fabric-samples/test-network$ peer chaincode query -o localhost:7050 --  
ordererTLSHostnameOverride orderer.example.com --tls --cafile "${PWD}/organizations/ordererOrganizations/example.com/order  
ers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n secured -c {"\\"function\\\":\"GetAssetPriva  
teProperties\"}, {"Args": [{"\"$ASSET_ID\": \""}]}  
{"object_type": "asset_properties", "color": "blue", "size": 35, "salt": "a94a8fe5ccb19ba61c4c0873d391e987982fbbd3"}
```



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan

Command 8:

```
$ peer chaincode query -o localhost:7050 --ordererTLSHostnameOverride
orderer.example.com --tls --cafile
"${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.
e xample.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n
secured -c "{\"function\":\"ReadAsset\",\"Args\":[\"$ASSET_ID\"]}"
```

Success message:

```
{"objectType":"asset","assetID":"d9923f21b770adbc79cbcc47a3aecc81dc7f0
30bd129155301ce3932be7fbcc","ownerOrg":"Org1MSP","publicDescription":" A
new asset for Org1MSP"}
```

```
subodh@subodh-HP-Laptop-15-bs0xx:~/hyperledger/prac2/fabric-samples/test-network$ peer chaincode query -o localhost:7050 --
ordererTLSHostnameOverride orderer.example.com --tls --cafile "${PWD}/organizations/ordererOrganizations/example.com/orderers/
orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n secured -c "{\"function\":\"ReadAsset\",
\"Args\":[\"$ASSET_ID\"]}"
{"objectType":"asset","assetID":"d9923f21b770adbc79cbcc47a3aecc81dc7f030bd129155301ce3932be7fbcc","ownerOrg":"Org1MSP",
"publicDescription":"A new asset for Org1MSP"}
```

Command 9:

```
$ peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride
orderer.example.com --tls --cafile
"${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.
e xample.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n
secured -c
"\"function\":\"ChangePublicDescription\",\"Args\":[\"$ASSET_ID\",\"This
asset is for sale\"]"
```

Success message:

```
2023-11-27 10:38:09.198 IST 0001 INFO [chaincodeCmd]
chaincodeInvokeOrQuery -> Chaincode invoke successful. result: status:200
```

```
subodh@subodh-HP-Laptop-15-bs0xx:~/hyperledger/prac2/fabric-samples/test-network$ peer chaincode invoke -o localhost:7050 -
-ordererTLSHostnameOverride orderer.example.com --tls --cafile "${PWD}/organizations/ordererOrganizations/example.com/orderers/
orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n secured -c "{\"function\":\"ChangePublic
Description\",\"Args\":[\"$ASSET_ID\"],\"This asset is for sale\"]}"
2023-11-27 10:38:09.198 IST 0001 INFO [chaincodeCmd] chaincodeInvokeOrQuery -> Chaincode invoke successful. result: status:
200
```



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan

Command 10:

```
$ peer chaincode query -o localhost:7050 --ordererTLSHostnameOverride
orderer.example.com --tls --cafile
"${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.
e xample.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n
secured -c "{\"function\":\"ReadAsset\",\"Args\":[\"$ASSET_ID\"]}"
```

Success message:

```
{"objectType":"asset","assetID":"d9923f21b770adbc79cbcc47a3aeccc81dc7f0
30bd129155301ce3932be7fbcc","ownerOrg":"Org1MSP","publicDescription":"
This asset is for sale"}
```

Command 11:

Operate from the Org2 terminal (bottom Terminal):

```
$ export
ASSET_ID=d9923f21b770adbc79cbcc47a3aeccc81dc7f030bd129155301ce39
32 be7fbcc

$ peer chaincode query -o localhost:7050 --ordererTLSHostnameOverride
orderer.example.com --tls --cafile
"${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.
e xample.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n
secured -c "{\"function\":\"ReadAsset\",\"Args\":[\"$ASSET_ID\"]}"
```

Success message:

```
{"objectType":"asset","assetID":"d9923f21b770adbc79cbcc47a3aeccc81dc7f0
30bd129155301ce3932be7fbcc","ownerOrg":"Org1MSP","publicDescription":"
This asset is for sale"}
```

Command 12:

```
$ peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride
orderer.example.com --tls --cafile
"${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.
e xample.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n
secured -c "{\"function\":\"ChangePublicDescription\",\"Args\":[\"$ASSET_ID\",
\"the
worst asset\"]}"
```

Success message:

```
2023-11-27 10:46:36.176 IST 0001 INFO [chaincodeCmd]
chaincodeInvokeOrQuery -> Chaincode invoke successful. result: status:200
```



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan

```
subodh@subodh-HP-Laptop-15-bs0xx:/hyperledger/prac2/fabric-samples/test-network$ peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls --cafile "${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n secured -c "{\"function\":\"\\\"ChangePublicDescription\\\"\", \"Args\":[\"$ASSET_ID\", \"the worst asset\"]}"
```

Command 13:

Agree to sell the asset

```
$ export ASSET_PRICE=$(echo -n
  "{\"asset_id\":\"$ASSET_ID\", \"trade_id\":\"109f4b3c50d7b0df729d299bc6f8e9ef9066971f\", \"price\":110} | base64 | tr -d \\n")
```

```
$ peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride
orderer.example.com --tls --cafile
"${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.
example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n
secured -c "{\"function\":\"\\\"AgreeToSell\\\"\", \"Args\":[\"$ASSET_ID\"]}" --transient
"\"asset_price\": \"$ASSET_PRICE\""
```

Success message:

```
2023-11-27 10:51:32.734 IST 0001 INFO [chaincodeCmd]
chaincodeInvokeOrQuery -> Chaincode invoke successful. result: status:200
```

```
subodh@subodh-HP-Laptop-15-bs0xx:/hyperledger/prac2/fabric-samples/test-network$ export ASSET_PRICE=$(echo -n "{\"asset_id\":\"$ASSET_ID\", \"trade_id\":\"109f4b3c50d7b0df729d299bc6f8e9ef9066971f\", \"price\":110} | base64 | tr -d \\n")
peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls --cafile "${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n
secured -c "{\"function\":\"\\\"AgreeToSell\\\"\", \"Args\":[\"$ASSET_ID\"]}" --transient "{\"asset_price\": \"$ASSET_PRICE\"}"
2023-11-27 10:52:32.736 IST 0001 INFO [chaincodeCmd] chaincodeInvokeOrQuery -> Chaincode invoke successful. result: status:200
```

Command 14:

```
$ peer chaincode query -o localhost:7050 --ordererTLSHostnameOverride
orderer.example.com --tls --cafile
"${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.
example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n
secured -c "{\"function\":\"\\\"GetAssetSalesPrice\\\"\", \"Args\":[\"$ASSET_ID\"]}"
```



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan
(Department of Computer Science)

Success message:

```
{"asset_id":"d9923f21b770adbc79cbcc47a3aeecc81dc7f030bd129155301ce39  
32be7fbcc","trade_id":"109f4b3c50d7b0df729d299bc6f8e9ef9066971f","pri  
ce":110}
```

```
subodh@subodh-HP-Laptop-15-bs0xx:/hyperledger/prac2/fabric-samples/test-network$ peer chaincode query -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls --cafile "${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n secured -c "{\"function\":\"GetAssetSalesPrice\"}","Args":["$ASSET_ID"]}"  
{"asset_id":"d9923f21b770adbc79cbcc47a3aeecc81dc7f030bd129155301ce3932be7fbcc","trade_id":"109f4b3c50d7b0df729d299bc6f8e9ef9066971f","price":110}
```

Command 15:

Update the asset description as Org2

```
$ peer chaincode query -o localhost:7050 --ordererTLSHostnameOverride  
orderer.example.com --tls --cafile  
"${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.  
.example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n  
secured -c  
"{"function":"GetAssetPrivateProperties","Args":["$ASSET_ID"]}"
```

Success message:

```
{"object_type":"asset_properties","color":"blue","size":35,"salt":"a94a8fe5cc  
b19ba61c4c0873d391e987982fbbd3"}
```

```
subodh@subodh-HP-Laptop-15-bs0xx:/hyperledger/prac2/fabric-samples/test-network$ peer chaincode query -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls --cafile "${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n secured -c "{\"function\":\"GetAssetPrivateProperties\"}","Args":["$ASSET_ID"]"}  
{"object_type":"asset_properties","color":"blue","size":35,"salt":"a94a8fe5ccb19ba61c4c0873d391e987982fbbd3"}
```

Command 16:

```
$ peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride  
orderer.example.com --tls --cafile  
"${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.e  
xample.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel  
-n secured -c  
"{"function":"ChangePublicDescription","Args":["$ASSET_ID","This  
asset is not for sale"]}"
```

Success message:

```
2023-11-27 11:18:30.055 IST 0001 INFO [chaincodeCmd]
```



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan
(Department of Computer Science)

chaincodeInvokeOrQuery -> Chaincode invoke successful. result: status:200

```
subodh@subodh-HP-Laptop-15-bs0xx:~/hyperledger/prac2/fabric-samples/test-network$ peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls --cafile "${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n secured -c "{\"function\":\"ChangePublicDescription\",\"Args\":[\"$ASSET_ID\",\"This asset is not for sale\"]}"  
2020-01-27 10:56:09.157 [INFO] [TMRV] [chaincodeCmd] chaincodeInvokeOrQuery -> Chaincode invoke successful. result: status: 200
```

Command 17:

```
$ peer chaincode query -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls --cafile "${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n secured -c "{\"function\":\"ReadAsset\",\"Args\":[\"$ASSET_ID\"]}"
```

Success message:

```
{"objectType":"asset","assetID":"d9923f21b770adbc79cbcc47a3aecc81dc70  
0  
30bd129155301ce3932be7fbcc","ownerOrg":"Org1MSP","publicDescription":  
" This asset is not for sale"}
```

```
subodh@subodh-HP-Laptop-15-bs0xx:~/hyperledger/prac2/fabric-samples/test-network$ peer chaincode query -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls --cafile "${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n secured -c "{\"function\":\"ReadAsset\",\"Args\":[\"$ASSET_ID\"]}"  
{"objectType":"asset","assetID":"d9923f21b770adbc79cbcc47a3aecc81dc7f030bd129155301ce3932be7fbcc","ownerOrg":"Org1MSP","pu  
blicDescription":"This asset is not for sale"}
```

Command 18:

Clean up (stopping chaincode):

```
$ ./network.sh down
```

```
Stopping containers  
4bd1008ea531  
797feba02da6  
Removing generated chaincode Docker Images  
Untagged: dev-peer0.org2.example.com-secured_1.0.1-843dc47073bc9ebd5af86d7f422fb794fc458d737c1fa4a1cb2b9cb2c4983d6b-cdfa6a2  
036afcda5b3a3f61997349ace1a5365ec66f47b9f97f59202eb4b4b:latest  
Deleted: sha256:a30d1940ef339af36a1b0e2b95abf2ad5baade214618e3df91753ebf57257cc  
Deleted: sha256:252d23302de4b0a256b550573bb2eae6980fa98a1a1332bed760db12b0e916  
Deleted: sha256:d0510e0bd4d0d963a2c42d43941f86a0fd613bad9660aa9f3eb9cabbb0aec  
Deleted: sha256:66bb9e10b5e19fad3886372e03764c36c53c7387e161351bbc85b32182d1be4  
Untagged: dev-peer0.org1.example.com-secured_1.0.1-843dc47073bc9ebd5af86d7f422fb794fc458d737c1fa4a1cb2b9cb2c4983d6b-76cd426  
ef69ed63b4d3b3a3d23c81abef151b4101749169b:latest  
Deleted: sha256:31169b1ee5bcce8da0f6f38f989e48970b42a53d6874b29281d17c932b62259  
Deleted: sha256:2484f45de4bf0ff0f6e950dbff49564213fc2ddda7fa3085c1356d0365f261b3  
Deleted: sha256:5ed3647a58b5761067be93c72862d3181254bd78c6ef448fd14808986b6b2a35  
Deleted: sha256:738bb199e6bfff95b4f7a0b5f6799188cece6046fe54123918977f6f3e9adb24b  
Unable to find image 'busybox:latest' locally  
latest: Pulling from library/busybox  
3f4d9098f5b: Pull complete  
Digest: sha256:3fbcc63216742a6d997e74f52b878d7cc478225cffac6bc977eedfe51c7f4e79  
Status: Downloaded newer image for busybox:latest
```



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan
(Department of Computer Science)

PRACTICAL 5

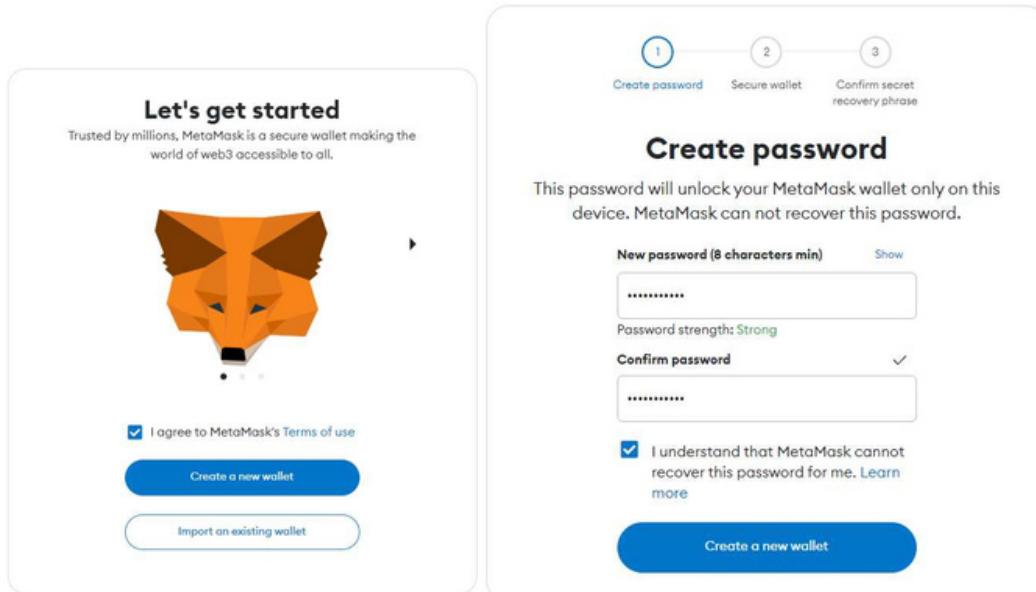
Aim: Set up MetaMask, set up Ganache, create two accounts with 100 ETH each, and transfer ETH between the accounts.

Background Information:

MetaMask is a cryptocurrency wallet and browser extension for Ethereum. It lets users store and manage Ethereum-based assets, interact with decentralized applications (DApps), and provides a secure way to access the Ethereum blockchain.

Install MetaMask Extension:

1. Go to the MetaMask website or the appropriate extension store for your browser (e.g., Chrome Web Store, Firefox Add-ons).
2. Add MetaMask to your browser and follow the on-screen instructions to create an account and set up a password. Save the seed phrase generated during the setup process in a secure location.





B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan
(Department of Computer Science)

The screenshots illustrate the setup process for a new Ethereum account using MetaMask. The first two panels show the initial 'Add account' configuration steps, while the third panel displays the main MetaMask interface for managing the account once it's created.

Setting up and Ganache:

The screenshots demonstrate the setup of a local Ethereum network using Ganache. The left image shows the download page for the Ganache application, and the right image shows the Ganache user interface displaying multiple accounts with initial balances.



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan
(Department of Computer Science)

Accessing Ganache Accounts:

After Ganache starts, it will display a list of accounts along with their private keys, addresses, and initial balances.

Import Accounts into MetaMask:

1. Open MetaMask in your browser and click on "Import Account."
2. Copy the private key of the first account from Ganache and import it into MetaMask by pasting the key.
3. Repeat the process for the second account.

Transferring ETH Between Accounts:

Access Ganache Accounts:

Note:

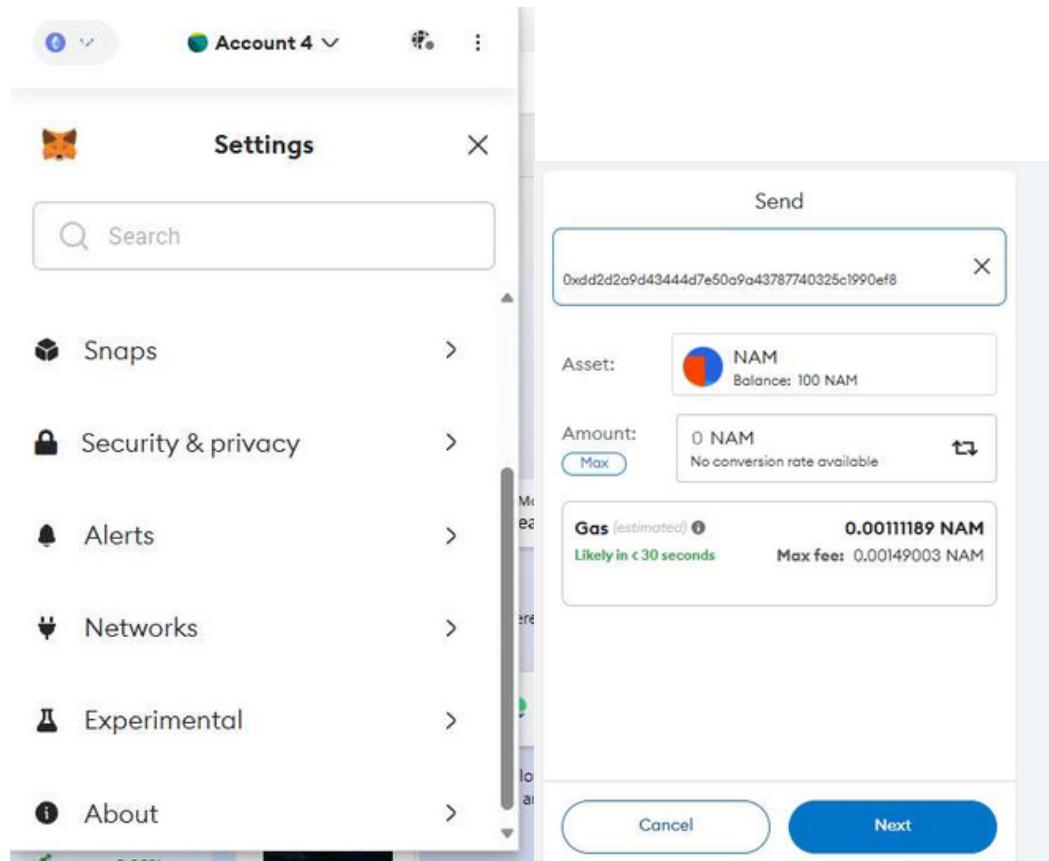
The addresses of the two accounts you imported into MetaMask from Ganache.

Transfer ETH from Account 1 to Account 2:

1. In MetaMask, select Account 1



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan
(Department of Computer Science)

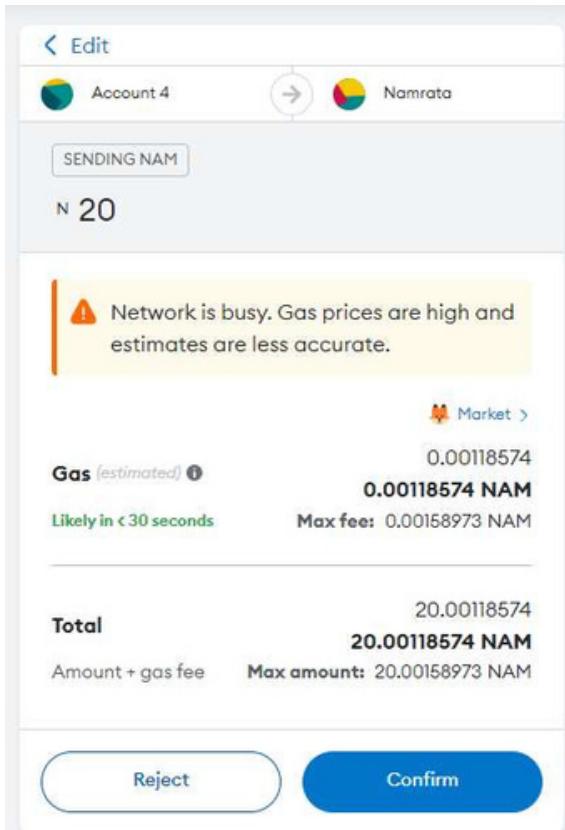


Click on "Send" or "Transfer" and enter the address of Account 2 in the recipient field.

2.



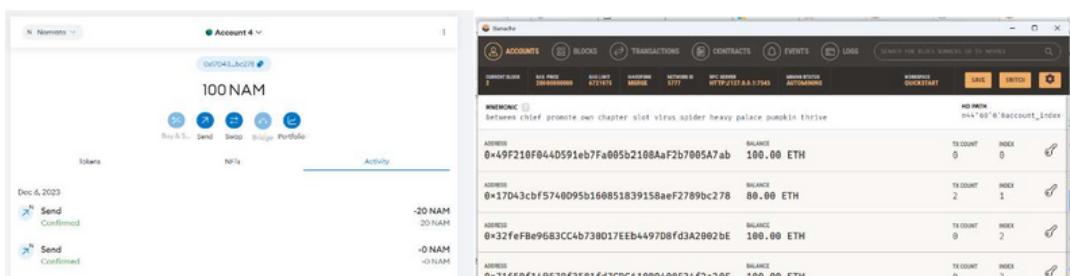
B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan
(Department of Computer Science)



3. Enter the amount of ETH you want to send and confirm the transaction.

4. Confirm the transaction in MetaMask by paying the gas fee.

5. Wait for the transaction to be confirmed on the local Ganache blockchain.





B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan
(Department of Computer Science)

PRACTICAL 6

Aim: setup and deploy Voting Smart contract in solidity programming language using Remix Ide.

Background Information:

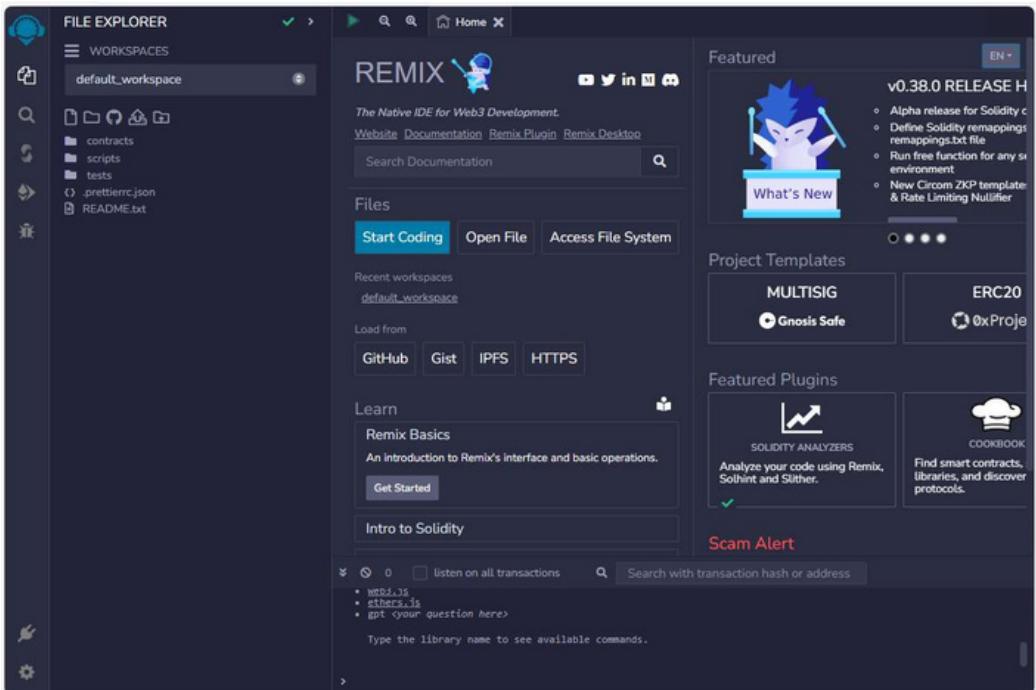
- Solidity : Programming language for Ethereum smart contracts.
- Remix IDE : Web tool for coding and deploying smart contracts.
- Smart Contracts : Self-executing agreements on the blockchain.
- SPDX-License-Identifier : Identifies the license for Solidity code.
- Mappings and Events : Key-value storage and logging in Solidity.
- Require Statement : Validates conditions; reverts if not met.
- Front-end Integration : Web interfaces for user interaction.
- Truffle and Hardhat : Frameworks for Ethereum development.

Access Remix IDE (<https://remix.ethereum.org/>) :

Open your web browser and go to Remix IDE. This is an online Solidity IDE provided by Ethereum.



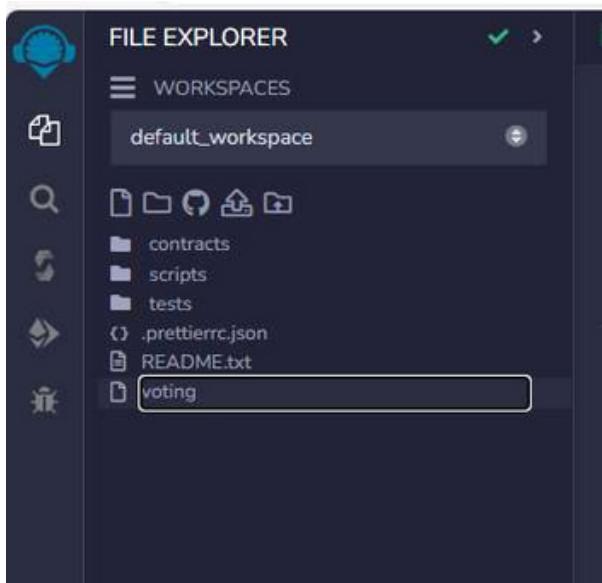
B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan
(Department of Computer Science)



Create a New File:

Click on the "file" icon on the left sidebar to create a new file in contract folder.

Name the file Voting.sol or any other relevant name for your voting contract.





B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan
(Department of Computer Science)

Write Your Smart Contract:

Code:

```
// SPDX-License-Identifier: MIT

pragma solidity ^0.8.0;


$$/* @title Storage$$


$$* @dev Store & retrieve value in a variable$$


$$* @custom:dev-run-script scripts/deploy_with_ETHERS.ts$$


$$*/$$


contract Voting {
    struct Candidate {
        uint id;
        string name;
        uint voteCount;
    }
    mapping(uint => Candidate) public candidates;
    mapping(address => bool) public hasVoted;
    uint public candidatesCount;
    constructor() {
```



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan
(Department of Computer Science)

```
addCandidate("Candidate 1");
addCandidate("Candidate 2");
}

function addCandidate(string memory _name) private {
candidatesCount++;
candidates[candidatesCount] = Candidate(candidatesCount, _name, 0);
}

function vote(uint _candidateId) public {
require(_candidateId > 0 && _candidateId <= candidatesCount, "Invalid candidate ID");
require(!hasVoted[msg.sender], "You have already voted");

hasVoted[msg.sender] = true;

candidates[_candidateId].voteCount++;

emit Voted(_candidateId, candidates[_candidateId].name);
}

event Voted(uint indexed candidateId, string candidateName);
}
```



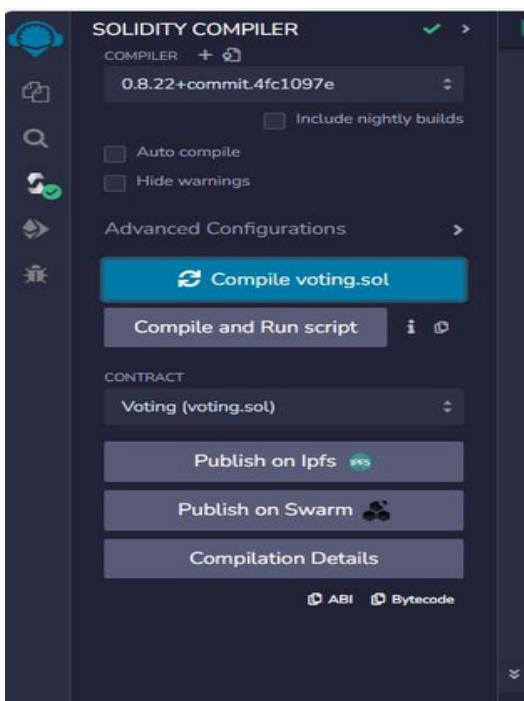
B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan
(Department of Computer Science)

The screenshot shows the Remix IDE interface. On the left is a file explorer with a workspace named "default_workspace" containing contracts, scripts, and tests. The main area displays the Solidity code for a "voting.sol" contract. The code defines a mapping of candidate names to vote counts, initializes two candidates, and handles the voting process. It includes a constructor, a vote function, and a Voted event. The code is annotated with gas estimates and compiler warnings.

```
string name;
uint voteCount;
}
mapping(uint => Candidate) public candidates;
mapping(address => bool) public hasVoted;
uint public candidatesCount;
constructor() {
    infinite gas 4202000 gas
    addCandidate("Candidate 1");
    addCandidate("Candidate 2");
}
uint256 internal _candidateId
name private {
    undefined gas
}
Estimated execution cost: infinite gas
function vote(uint _candidateId) public {
    infinite gas
    require(_candidateId > 0 && _candidateId <= candidatesCount, "Invalid candidate ID");
    require(!hasVoted[msg.sender], "You have already voted");
    hasVoted[msg.sender] = true;
    candidates[_candidateId].voteCount++;
    emit Voted(_candidateId, candidates[_candidateId].name);
}
event Voted(uint indexed candidateId, string candidateName);
```

Compiler Settings:

On the Remix interface, go to the "Solidity Compiler" tab located on the left sidebar. Choose the appropriate compiler version based on your smart contract code. Click "Compile" to compile your contract.





B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan
(Department of Computer Science)

Deploying and Interacting with the Smart Contract:

Deploy the Contract:

Switch to the "Deploy & Run Transactions" tab in Remix.

Ensure the correct environment is selected (e.g., JavaScript VM, Injected Web3, etc.).

Click on "Deploy" to deploy your contract. You'll get a deployed contract address.

```
string name;
uint voteCount;
}
mapping(uint => Candidate) public candidates;
mapping(address => bool) public hasVoted;
uint public candidatesCount;
constructor() {
    infinite gas 420200 gas
    addCandidate("Candidate 1");
    addCandidate("Candidate 2");
}

function addCandidate(string memory _name) private {
    undefined gas
    candidatesCount++;
    candidates[candidatesCount] = Candidate(candidatesCount, _name, 0);
}

function vote(uint _candidateId) public {
    infinite gas
    require(_candidateId > 0 && _candidateId <= candidatesCount, "Invalid candidate ID");
    require(!hasVoted[msg.sender], "You have already voted");

    hasVoted[msg.sender] = true;
    candidates[_candidateId].voteCount++;
    emit Voted(_candidateId, candidates[_candidateId].name);
}

event Voted(uint indexed candidateId, string candidateName);
}
```



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan
(Department of Computer Science)

PRACTICAL 7

Aim: Build a decentralized Lottery application (DApp), Using ERC-20 Tokens that combines Ethereum's Web3 and Solidity smart contracts deploy on Test network.

Background Information:

To create a decentralized lottery DApp:

1. Smart Contract (Solidity): Write a Solidity smart contract handling ticket purchases, random number generation, and prize distribution.
2. Deployment: Deploy the smart contract on an Ethereum test network.
3. Frontend (Web3.js): Develop a frontend interface using Web3.js to interact with the smart contract. Enable users to buy lottery tickets and trigger the random winner selection.

Steps to follow:

Step 1: Remix IDE (<https://remix.ethereum.org/>).

- It's an online Solidity IDE provided by Ethereum, where you can write, compile, and deploy Solidity smart contracts.

Step 2: Creating a New File

- Click on the "+" icon on the left-hand side or the "File" menu at the top to create a new file.
- Name the file with a .sol extension, for example, Lottery.sol.
- Write your Solidity code into this file.

Code:

```
// SPDX-License-Identifier: GPL-3.0
```

```
pragma solidity ^0.8.11;
/**
 * @title Storage
 * @dev Store & retrieve value in a variable
 * @custom:dev-run-script ./scripts/deploy_with_ETHERS.ts
 */
contract Lottery {
    address public owner;
```



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan
(Department of Computer Science)

```
address payable[] public players;
uint public lotteryId;
mapping (uint => address payable) public lotteryHistory;

constructor() {
    owner = msg.sender;
    lotteryId = 1;
}

function getWinnerByLottery(uint lottery) public view returns (address payable) {
    return lotteryHistory[lottery];
}

function getBalance() public view returns (uint) {
    return address(this).balance;
}

function getPlayers() public view returns (address payable[] memory) {
    return players;
}

function enter() public payable {
    require(msg.value > 1 ether);
    // address of player entering lottery
    players.push(payable(msg.sender));
}

function getRandomNumber() public view returns (uint) {
    return uint(keccak256(abi.encodePacked(owner, block.timestamp)));
}

function pickWinner() public onlyowner {
    uint index = getRandomNumber() % players.length;
    players[index].transfer(address(this).balance);

    lotteryHistory[lotteryId] = players[index];
    lotteryId++;
}

players = new address payable[](0);
}

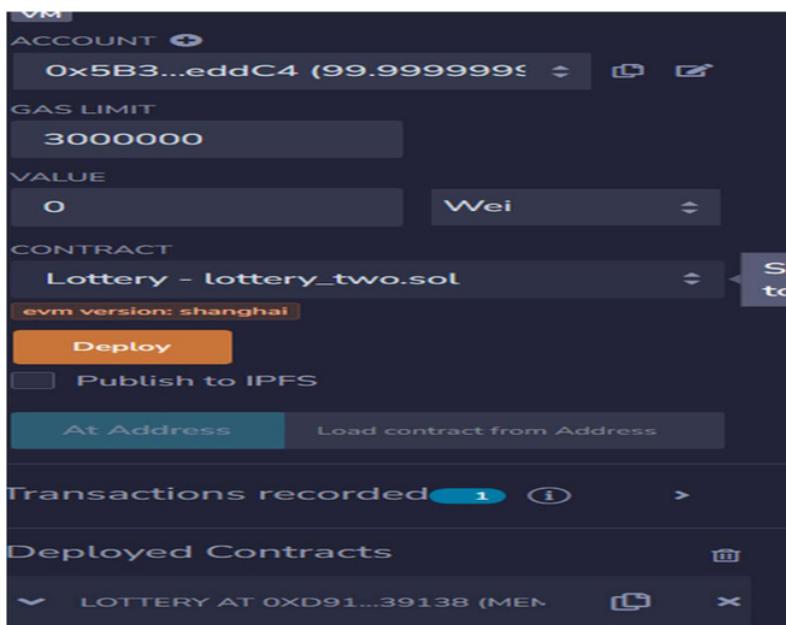
modifier onlyowner() {
    require(msg.sender == owner);
```



```
};  
}  
}
```

Step 3: Compiling and Run the Contract

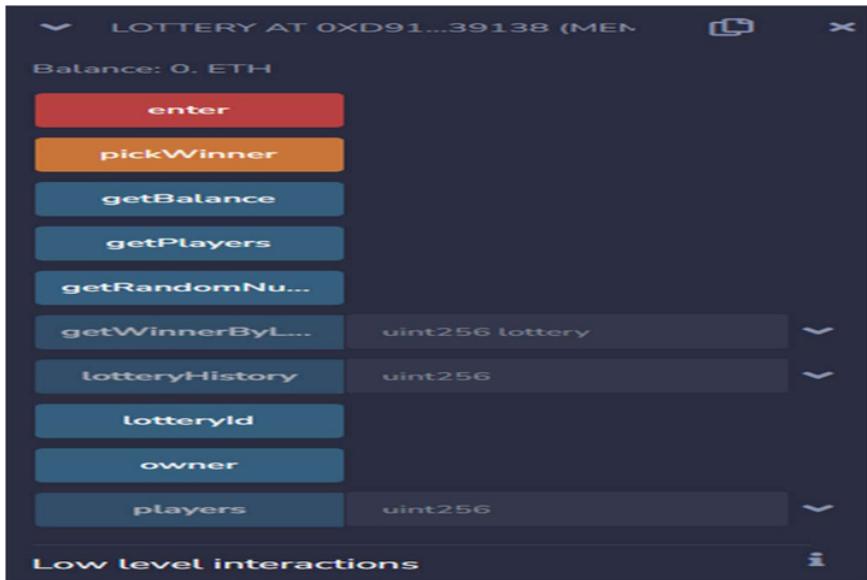
- On the left-hand side panel, go to the "Solidity Compiler" tab (it looks like a stack of papers).
- Ensure that the appropriate Solidity compiler version is selected.
- Click on "Compile Lottery.sol" (or the name of your file) to compile the contract.
- Verify that there are no errors in the compilation output panel.
- Click on deploy for first account to set it as manager.
- Then select another account set value to 2 Ether.
- Click on Transact button.



- Repeat more than equal to 3 times to set the number of participants to create Transaction pool for lottery. Remix will display transaction details and logs when you perform actions like deploying a contract or calling its functions. You can also check the transaction hash, gas usage, and status of transactions executed on the Ethereum blockchain.



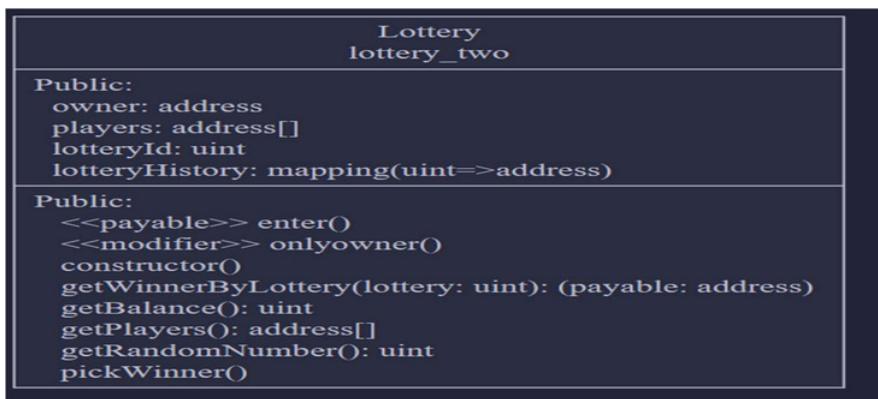
B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan
(Department of Computer Science)



- You can see the balance of the Transaction pool as manager • Also, can see the transactions executed successfully. • To Execute lottery, click on pickwinner button as manager the winner will be picker from all the participants.

```
[✓] [vm] from: 0x482...C02db to: Lottery.enter() 0xd8b...33fa8 value: 20000000000000000000 wei data: 0xe97...dcb62 logs: 0 hash: 0x859...3ic0a Debug ▾  
transact to Lottery.enter pending ...  
  
[✓] [vm] from: 0x787...cabab to: Lottery.enter() 0xd8b...33fa8 value: 20000000000000000000 wei data: 0xe97...dcb62 logs: 0 hash: 0xc40...b5de3 Debug ▾  
transact to Lottery.enter pending ...  
  
[✓] [vm] from: 0x617...5E7f2 to: Lottery.enter() 0xd8b...33fa8 value: 20000000000000000000 wei data: 0xe97...dcb62 logs: 0 hash: 0x56b...57b9a Debug ▾  
transact to Lottery.pickWinner pending ...  
  
[✓] [vm] from: 0x5B3...eddC4 to: Lottery.pickWinner() 0xd8b...33fa8 value: 0 wei data: 0x5d4...95aea logs: 0 hash: 0x0ec...10eb3 Debug ▾
```

UML diagram to show the architecture, design, and implementation of our Lottery DApp.





B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan
(Department of Computer Science)

PRACTICAL 8

Aim: Practical on Supply Chain Dapp.

Background Information:

A Supply Chain DApp leverages blockchain technology and smart contracts to enhance the efficiency, transparency, and security of supply chain processes. Key components include product traceability, decentralized supplier management, real-time inventory tracking, quality assurance with immutable records, and transparent logistics operations. Smart contracts automate various aspects, such as agreements, reordering, and quality checks, ensuring a reliable and tamper-resistant supply chain.

Steps to follow:

Step 1:

Go to Remix IDE (<https://remix.ethereum.org/>).

- It's an online Solidity IDE provided by Ethereum, where you can write, compile, and deploy Solidity smart contracts.

Step 2: Creating a New File

- Click on the "+" icon on the left-hand side or the "File" menu at the top to create a new file.
- Name the file with a .sol extension, for example, Supplychain.sol.
- Write your Solidity code into this file.

Code:

```
// SPDX-License-Identifier: GPL-3
pragma solidity ^0.8.0;

/**
 * @title Storage
 * @dev Store & retrieve value in a variable
 * @custom:dev-run-script ./scripts/deploy_with_ETHERS.ts
```



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan
(Department of Computer Science)

*/

```
contract SupplyChain {
    uint public productCount = 0;

    struct Product {
        uint id;
        string name;
        uint quantity;
        address owner;
        address payable[] history;
    }

    mapping(uint => Product) public products;

    event ProductCreated(uint id, string name, uint quantity, address owner);
    event ProductTransferred(uint id, address from, address to);

    function createProduct(string memory _name, uint _quantity) public {
        productCount++;
        address payable[] memory initialHistory;
        products[productCount] = Product(productCount, _name, _quantity, msg.sender,
                                         initialHistory);
        emit ProductCreated(productCount, _name, _quantity, msg.sender);
    }

    function transferProduct(uint _productId, address _newOwner) public {
        require(_productId > 0 && _productId <= productCount, "Invalid ID");
    }
}
```



B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan
(Department of Computer Science)

```
Product storage product = products[_productId];
require(msg.sender == product.owner, "Only the owner can transfer the product");

product.owner = _newOwner;
product.history.push(payable(_newOwner));

emit ProductTransferred(_productId, msg.sender, _newOwner);
}

function getProductHistory(uint _productId) public view returns (address payable[])
memory {
require(_productId > 0 && _productId <= productCount, "Invalid product ID");
return products[_productId].history;
}
}
```

Step 3: Compiler Settings

On the Remix interface, go to the "Solidity Compiler" tab located on the left sidebar.

Choose the appropriate compiler version based on your smart contract code.

Click "Compile" to compile your contract.



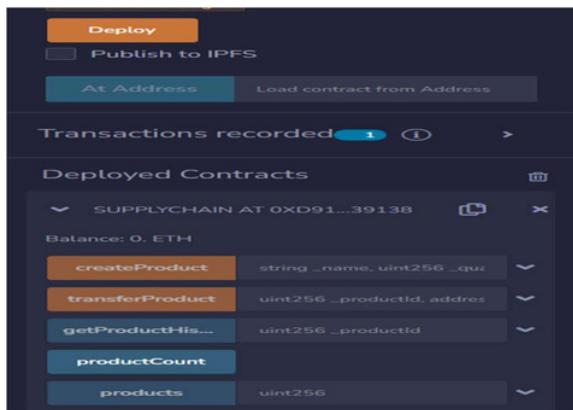


B. K. Birla College of Arts, Science & Commerce (Autonomous), Kalyan
(Department of Computer Science)

Step 4: Deploy the Contract

Switch to the "Deploy & Run Transactions" tab in Remix.

Click on "Deploy" to deploy your contract. You'll get a deployed contract address.



Step 5:

Click on createProduct and add your product name in double quotes for example “cake” and then add quantity of the product.

Click on transact button you will get succeed transaction address and then call product.

