Instructions for starting the Blockchain & deploying Smart Contract on top of it

1. The *kyc-truffle-geth* folder contains two sub-folders:
    i. *kyc-smart-contract-truffle* (This is created with truffle and contains solidity files for the smart contract and other deployment related & config files.)
    ii. *kyc-chaindata-geth* (This contains the genesis file (init.json) for starting private Blockchain network, over which our **AdminApp** contract will be deployed.)
2. Start the terminal and make sure that you are into the *kyc-chaindata-geth* directory.
3. Then, to create a private chain data for our Blockchain run the following command:

    geth --datadir ./datadir init ./init.json

4. Now, to start this network, type the following command at the same terminal window:

    geth --datadir ./datadir --networkid 2050 --nodiscover console

5. This will create a new blockchain network and it will start a console with which you can access the different APIs that are available for this private blockchain network you have created.
6. Now, let's create an account for admin (say password is 'admin') by typing the following command:

    personal.newAccount('admin')

7. Similarly, let's create 5 more accounts for 5 different banks, which admin will add, once we will deploy our **AdminApp** smart contract over this network. For this, say we are going to keep the passwords as 'bank1', 'bank2', ……..'bank5'

    personal.newAccount('bank1')

    run the same command for bank2, bank3 and so on upto bank5 to create five different accounts for banks.

8. Now, just to check the accounts, run the following command

    eth.accounts

    this will return a list of account addresses where, the first one we have designated for admin and rest 5 have been kept for adding 5 different banks by admin after **AdminApp** contract deployment.
    After this, you may come out of the network for now by pressing ctrl+d and press Enter

9. Now, we will start a new terminal window and go to the *kyc-smart-contract-truffle* directory inside this terminal.
10. Then, to start a new truffle project here, type the following command

    truffle init

    When you run this command Truffle prepares a framework of files that you would require to be able to successfully compile, migrate and test your smart contract projects using the Truffle environment.

11. Now, copy the same two solidity files namely **BankInterface.sol** and **KYC.sol** in the **contracts** sub-folder, which was created after running **truffle init** command.
12. Now, inside the **migrations** sub-folder create a new file ***2_kyc_migration.js*** to deploy **AdminApp** smart contract on the private Blockchain network, which we created earlier. *(necessary migration script has been written in this js file).*
13. Now, go to the ***truffle-config.js*** file and define the various network parameters and compiler version (solc compiler), your smart contract is using. *(necessary configurations have already been defined.)*
14. After this, run the following command at the same terminal window:

    truffle compile

    after running this command, you can see that a new sub-folder 'build' has been created, which is containing compiled version of all the contracts defined in our solidity files.
15. Now, go back to other terminal window, where we set up our private Blockchain network and start the network with following command:

    geth --datadir ./datadir --networkid 2050 --nodiscover --http --http.port 30303 --allow-insecure-unlock console

    after pressing enter, your geth environment is up and running.

16. Now at the same terminal, run the following command to unlock first account (admin account), so that it is accessible to the truffle console environment when you want to send your smart contracts on the Blockchain:

    personal.unlockAccount('<address of 1st account>', 'password for this account', 0)

    after, you press enter, true is returned, which means the account is unlocked.
17. Now start the mining process by running following command:

    miner.start()

    after pressing enter, you will see that the mining has been started in our private Blockchain network and now our network is ready to accept the truffle connection and install the **AdminApp** smart contract.
18. Now, go to the terminal, where we had our truffle environment and run following command:

    truffle migrate --network development

    after pressing enter, our smart contract **AdminApp** is now installed on the private Blockchain network.
19. Now, we just have to use the truffle console environment to get an instance of the **AdminApp** smart contract and test various functions of it.
20. To access the truffle console, run following command:

    truffle console --network development

    after pressing enter, you are now inside the truffle console connected to the geth environment.

21. To get an instance of the **AdminApp** smart contract, write the following javascript statement inside the truffle console:

    let kyc = await AdminApp.deployed()

    when this is executed, we will have instance of **AdminApp** smart contract inside the variable **kyc.**
22. Now, when you write **kyc.** and press **Tab twice**, you will be able to see all methods that are available to this instance of **AdminApp** smart contract.
23. Now, you can test all methods using this **kyc** variable.
    For example, say admin wants to add a bank, then it can be done by writing the following:

    kyc.addBank(“<bank name>”, “0x<hex address of the bank”, “registration number”)

24. Other methods as mentioned in the problem statement may also be tested in the same manner.