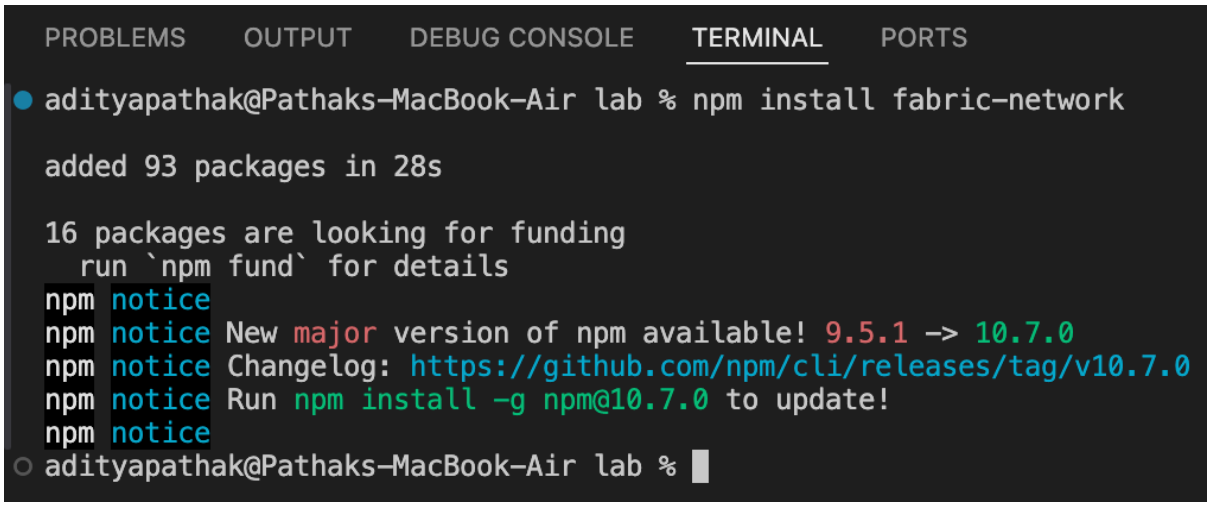
Blockchain Lab

Assignment-7

Name: Aditya Pathak
Subgroup: 3NC2
Roll Number: 102115044

The Hyperledger Fabric Software Development Kit (SDK) enables developers to build a wide range of applications for enterprise blockchain solutions.

Here's a simplified example of an identity management application using the Hyperledger Fabric SDK in Node.js. This example demonstrates how to enroll users, register new users, and perform identity-related operations on a Hyperledger Fabric network.



Installed Hyperledger Fabric Software Development Kit

Using VS Code Terminal

Code

JS hyperledgerFabric_task7.js × {} package-lock.json

JS hyperledgerFabric_task7.js > ...

```
const { Wallets, Gateway } = require('fabric-network');
const FabricCAServices = require('fabric-ca-client');

async function main() {
    const gateway = new Gateway();

    try {
        const wallet = await Wallets.newFileSystemWallet('./wallet');

        const caURL = 'http://localhost:7054';
        const ca = new FabricCAServices(caURL);

        const adminIdentity = await wallet.get('admin');
        if (!adminIdentity) {
            console.log('An identity for the admin user does not exist in the wallet');
            return;
        }

        const gatewayOptions = {
            identity: 'admin',
            wallet: wallet,
            discovery: { enabled: true, asLocalhost: true }
        };

        await gateway.connect('mychannel', gatewayOptions);

        const network = await gateway.getNetwork('mychannel');

        const contract = network.getContract('basic');

        // enrolling a user
        const enrollment = await ca.enroll({ enrollmentID: 'user1', enrollmentSecret: 'user1pw' });
        const userIdentity = {
            credentials: {
                certificate: enrollment.certificate,
                privateKey: enrollment.key.toBytes(),
            },
            mspId: 'Org1MSP',
            type: 'X.509',
        };

        await wallet.put('user1', userIdentity);

        console.log('Successfully enrolled user1 and imported it into the wallet');
```

```
45
46          // registering a new user
47          const registerRequest = {
48              enrollmentID: 'user2',
49              affiliation: 'org1.department1',
50              role: 'client',
51          };
52
53          const secret = await ca.register(registerRequest, adminIdentity);
54          console.log(`Successfully registered user2 with enrollment secret: ${secret}`);
55
56      } catch (error) {
57          console.error(`Error: ${error}`);
58          process.exit(1);
59      } finally {
60          gateway.disconnect();
61      }
62  }
63
64  main();
```