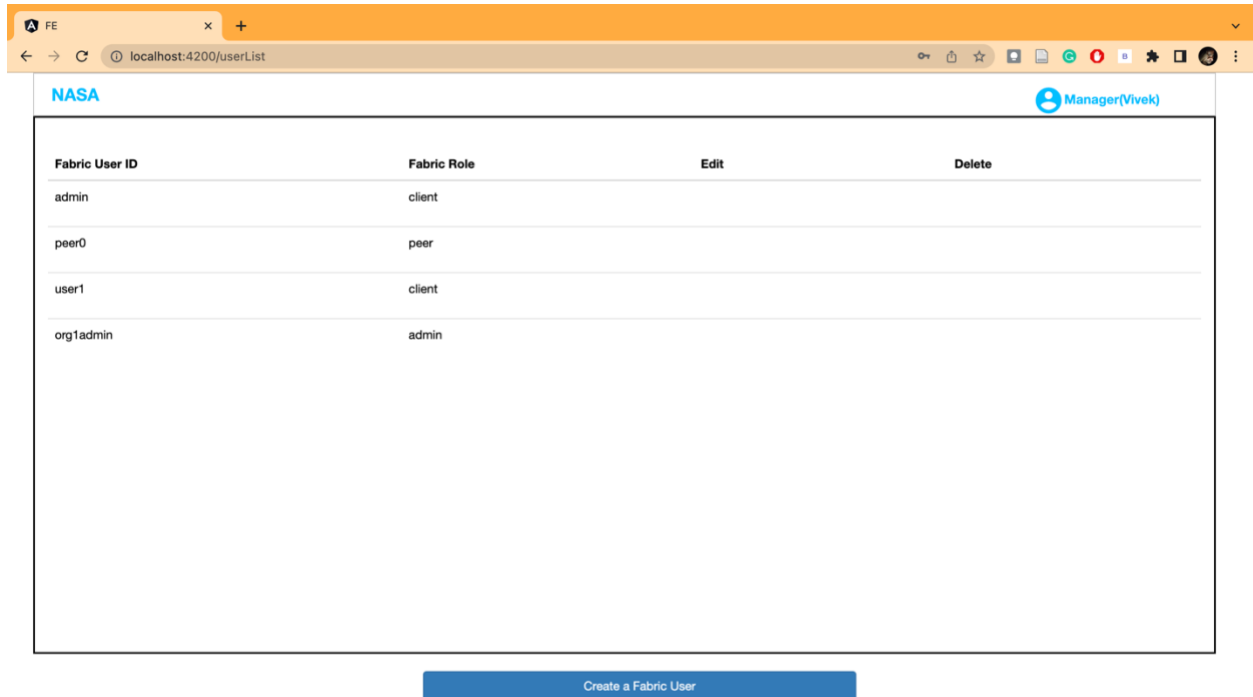


CRUD Operations

The fabric user's identity list in the UI

- It will show the all the users that are registered in fabric ca (Including admins)



The screenshot shows a web browser window with the URL `localhost:4200/userList`. The application has a header with the NASA logo on the left and a user profile 'Manager(Vivek)' on the right. Below the header is a table with the following columns: 'Fabric User ID', 'Fabric Role', 'Edit', and 'Delete'. The table contains four rows of user data:

Fabric User ID	Fabric Role	Edit	Delete
admin	client		
peer0	peer		
user1	client		
org1admin	admin		

Below the table is a blue button labeled 'Create a Fabric User'.

```
~/Desktop/Capstone_Application_2022/fabric-samples/fabric -- zsh
vivekkukkapalli@Viveks-MacBook-Pro fabric-samples % bin/fabric-ca-client identity list
Name: admin, Type: client, Affiliation: , Max Enrollments: -1, Attributes: [{(Name:hf.GenCRL Value:1 ECert:false) (Name:hf.Registrar.Attributes Value:* ECert:false) (Name:hf.AffiliationMgr Value:1 ECert:fa
lse) (Name:hf.Registrar.Roles Value:* ECert:false) (Name:hf.Registrar.DelegateRoles Value:* ECert:false) (Name:hf.Revoker Value:1 ECert:false) (Name:hf.IntermediateCA Value:1 ECert:false)}]
Name: peer0, Type: peer, Affiliation: , Max Enrollments: -1, Attributes: [{(Name:hf.EnrollmentID Value:peer0 ECert:true) (Name:hf.Type Value:peer ECert:true) (Name:hf.Affiliation Value: ECert:true)}]
Name: user1, Type: client, Affiliation: , Max Enrollments: -1, Attributes: [{(Name:hf.EnrollmentID Value:user1 ECert:true) (Name:hf.Type Value:client ECert:true) (Name:hf.Affiliation Value: ECert:true)}]
Name: org1admin, Type: admin, Affiliation: , Max Enrollments: -1, Attributes: [{(Name:hf.EnrollmentID Value:org1admin ECert:true) (Name:hf.Type Value:admin ECert:true) (Name:hf.Affiliation Value: ECert:tru
e)}]
vivekkukkapalli@Viveks-MacBook-Pro fabric-samples %
```

Creating a fabric User Identity

- To create a new fabric user, click on **Create a Fabric User** Button



Create an Fabric User

User Identity

Role

Affiliation

Register & Enroll

- Creating a new fabric user identity SE1 without selecting any attribute values for Role & Affiliation. Role can be either (peer or client) for now.



Create an Fabric User

User Identity

Role

☒ peer
☐ client
☐ admin

Register & Enroll

- Once details are entered click on **Register & Enroll** to register and enroll the fabric user identity into fabric ca.

- After successful enrollment the identity will be reflected in the Front End
- If no value is selected for the role option, by default value (Client) will be assigned

Fabric User ID	Fabric Role	Edit	Delete
admin	client		
peer0	peer		
user1	client		
org1admin	admin		
SE1	client		

Create a Fabric User

Successfully registered and enrolled SE1 OK

```



~/Desktop/Capstone_Application_2022/fabric-samples/fabric-ca --zsh
vivekkukkapalli@Viveks-MacBook-Pro fabric-samples % bin/fabric-ca-client identity list
Name: admin, Type: client, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.GenCRL Value:1 ECert:false) (Name:hf.Registrar.Attributes Value:* ECert:false) (Name:hf.AffiliationMgr Value:1 ECert:fa
lse) (Name:hf.Registrar.Roles Value:* ECert:false) (Name:hf.Registrar.DelegateRoles Value:* ECert:false) (Name:hf.Revoker Value:1 ECert:false) (Name:hf.IntermediateCA Value:1 ECert:false)]
Name: peer0, Type: peer, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:peer0 ECert:true) (Name:hf.Type Value:peer ECert:true) (Name:hf.Affiliation Value: ECert:true)]
Name: user1, Type: client, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:user1 ECert:true) (Name:hf.Type Value:client ECert:true) (Name:hf.Affiliation Value: ECert:true)]
Name: org1admin, Type: admin, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:org1admin ECert:true) (Name:hf.Type Value:admin ECert:true) (Name:hf.Affiliation Value: ECert:tru
e)]
vivekkukkapalli@Viveks-MacBook-Pro fabric-samples % bin/fabric-ca-client identity list
Name: admin, Type: client, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.GenCRL Value:1 ECert:false) (Name:hf.Registrar.Attributes Value:* ECert:false) (Name:hf.AffiliationMgr Value:1 ECert:fa
lse) (Name:hf.Registrar.Roles Value:* ECert:false) (Name:hf.Registrar.DelegateRoles Value:* ECert:false) (Name:hf.Revoker Value:1 ECert:false) (Name:hf.IntermediateCA Value:1 ECert:false)]
Name: peer0, Type: peer, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:peer0 ECert:true) (Name:hf.Type Value:peer ECert:true) (Name:hf.Affiliation Value: ECert:true)]
Name: user1, Type: client, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:user1 ECert:true) (Name:hf.Type Value:client ECert:true) (Name:hf.Affiliation Value: ECert:true)]
Name: org1admin, Type: admin, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:org1admin ECert:true) (Name:hf.Type Value:admin ECert:true) (Name:hf.Affiliation Value: ECert:tru
e)]
Name: SE1, Type: client, Affiliation: , Max Enrollments: 1, Attributes: [(Name:hf.EnrollmentID Value:SE1 ECert:true) (Name:hf.Type Value:client ECert:true) (Name:hf.Affiliation Value: ECert:true)]
vivekkukkapalli@Viveks-MacBook-Pro fabric-samples %

```

Updating the Fabric User Identity

- Updating the role of the fabric user identity SE1 to peer
- To edit the role of the user identity, click on edit option
- admin identities cannot be edited

The screenshot shows a web browser window with the URL `localhost:4200/userList`. The page header includes the NASA logo and a user profile for 'Manager(Vivek)'. The main content is a table with the following data:

Fabric User ID	Fabric Role	Edit	Delete
admin	client		
peer0	peer		
user1	client		
org1admin	admin		
SE1	client		

Below the table, there is a blue button labeled 'Create a Fabric User'.

Editing page

- Changing the role to the peer

FE x +

localhost:4200/updateFabricUser;userInfo=%7B%22id%22%3A%22SE1%22%2C%22type%22%3A%22client%22%2C%22affiliation%22%3A%22%7D

NASA Manager(Vivek)

Update Fabric User

User Identity

SE1

Role

peer

Affiliation



Update

Role has been updated in the user list page in the front end

FE x +

localhost:4200/userList

NASA Manager(Vivek)

Fabric User ID	Fabric Role	Edit	Delete
admin	client		
peer0	peer		
user1	client		
org1admin	admin		
SE1	peer		

Create a Fabric User

Successfully updated details of fabric user SE1 OK

```
~/Desktop/Capstone_Application_2022/fabric-samples/fabric -- zsh ... ~/Desktop/Capstone_Application_2022/fabric-samples -- zsh +
vivekkukkapalli@Viveks-MacBook-Pro fabric-samples % bin/fabric-ca-client identity list
Name: admin, Type: client, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.GenCRL Value:1 ECert:false) (Name:hf.Registrar.Attributes Value:* ECert:false) (Name:hf.AffiliationMgr Value:1 ECert:fa
lse) (Name:hf.Registrar.Roles Value:* ECert:false) (Name:hf.Registrar.DelegateRoles Value:* ECert:false) (Name:hf.Revoker Value:1 ECert:false) (Name:hf.IntermediateCA Value:1 ECert:false)]
Name: peer0, Type: peer, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:peer0 ECert:true) (Name:hf.Type Value:peer ECert:true) (Name:hf.Affiliation Value: ECert:true)]
Name: user1, Type: client, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:user1 ECert:true) (Name:hf.Type Value:client ECert:true) (Name:hf.Affiliation Value: ECert:true)]
Name: org1admin, Type: admin, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:org1admin ECert:true) (Name:hf.Type Value:admin ECert:true) (Name:hf.Affiliation Value: ECert:tru
e)]
vivekkukkapalli@Viveks-MacBook-Pro fabric-samples % bin/fabric-ca-client identity list
Name: admin, Type: client, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.GenCRL Value:1 ECert:false) (Name:hf.Registrar.Attributes Value:* ECert:false) (Name:hf.AffiliationMgr Value:1 ECert:fa
lse) (Name:hf.Registrar.Roles Value:* ECert:false) (Name:hf.Registrar.DelegateRoles Value:* ECert:false) (Name:hf.Revoker Value:1 ECert:false) (Name:hf.IntermediateCA Value:1 ECert:false)]
Name: peer0, Type: peer, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:peer0 ECert:true) (Name:hf.Type Value:peer ECert:true) (Name:hf.Affiliation Value: ECert:true)]
Name: user1, Type: client, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:user1 ECert:true) (Name:hf.Type Value:client ECert:true) (Name:hf.Affiliation Value: ECert:true)]
Name: org1admin, Type: admin, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:org1admin ECert:true) (Name:hf.Type Value:admin ECert:true) (Name:hf.Affiliation Value: ECert:tru
e)]
Name: SE1, Type: client, Affiliation: , Max Enrollments: 1, Attributes: [(Name:hf.EnrollmentID Value:SE1 ECert:true) (Name:hf.Type Value:client ECert:true) (Name:hf.Affiliation Value: ECert:true)]
vivekkukkapalli@Viveks-MacBook-Pro fabric-samples % bin/fabric-ca-client identity list
Name: admin, Type: client, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.GenCRL Value:1 ECert:false) (Name:hf.Registrar.Attributes Value:* ECert:false) (Name:hf.AffiliationMgr Value:1 ECert:fa
lse) (Name:hf.Registrar.Roles Value:* ECert:false) (Name:hf.Registrar.DelegateRoles Value:* ECert:false) (Name:hf.Revoker Value:1 ECert:false) (Name:hf.IntermediateCA Value:1 ECert:false)]
Name: peer0, Type: peer, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:peer0 ECert:true) (Name:hf.Type Value:peer ECert:true) (Name:hf.Affiliation Value: ECert:true)]
Name: user1, Type: client, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:user1 ECert:true) (Name:hf.Type Value:client ECert:true) (Name:hf.Affiliation Value: ECert:true)]
Name: org1admin, Type: admin, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:org1admin ECert:true) (Name:hf.Type Value:admin ECert:true) (Name:hf.Affiliation Value: ECert:tru
e)]
Name: SE1, Type: peer, Affiliation: , Max Enrollments: 1, Attributes: [(Name:hf.EnrollmentID Value:SE1 ECert:true) (Name:hf.Type Value:peer ECert:true) (Name:hf.Affiliation Value: ECert:true)]
vivekkukkapalli@Viveks-MacBook-Pro fabric-samples %
```

Deleting the User Identity

- To delete an user identity click on delete icon assigned to the respective user identity
- Need to select a reason, why you want to delete an user identity

The screenshot shows a web browser window with the URL `localhost:4200/userList`. The page displays a table of Fabric users. The user 'SE1' is highlighted with a red box around the delete icon.

Fabric User ID	Fabric Role	Edit	Delete
admin	client		
peer0	peer		
user1	client		
org1admin	admin		
SE1	peer		

At the bottom of the page, there is a blue button labeled "Create a Fabric User".



Delete Fabric User

User Identity

Reason

Delete



Delete Fabric User

User Identity

Reason

✓ unspecified

keycompromise

cacompromise

affiliationchanged

superseded

cessationofoperation

certificatehold

removefromcrl

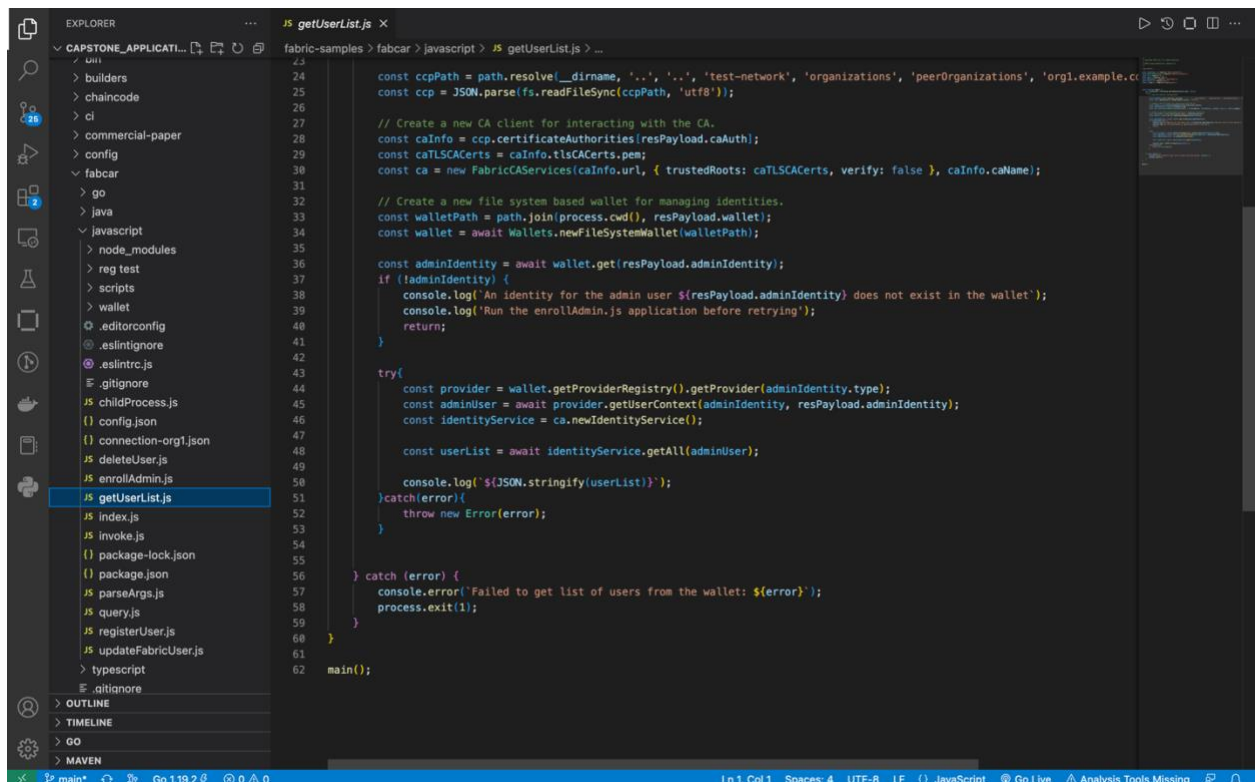
privilegewithdrawn

aacompromise


```
~/Desktop/Capstone_Application_2022/fabric-samples/fabcar --zsh
vivekkukkapalli@Viveks-MacBook-Pro fabric-samples % bin/fabric-ca-client identity list
Name: admin, Type: client, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.GenCRL Value:1 ECert:false) (Name:hf.Registrar.Attributes Value: ECert:false) (Name:hf.AffiliationMgr Value:1 ECert:fa
lse) (Name:hf.Registrar.Roles Value: ECert:false) (Name:hf.Registrar.DelegateRoles Value: ECert:false) (Name:hf.Revoker Value:1 ECert:false) (Name:hf.IntermediateCA Value:1 ECert:false)]
Name: peer0, Type: peer, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:peer0 ECert:true) (Name:hf.Type Value:peer ECert:true) (Name:hf.Affiliation Value: ECert:true)]
Name: user1, Type: client, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:user1 ECert:true) (Name:hf.Type Value:client ECert:true) (Name:hf.Affiliation Value: ECert:true)]
Name: orgladmin, Type: admin, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:orgladmin ECert:true) (Name:hf.Type Value:admin ECert:true) (Name:hf.Affiliation Value: ECert:tru
e)]
vivekkukkapalli@Viveks-MacBook-Pro fabric-samples % bin/fabric-ca-client identity list
Name: admin, Type: client, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.GenCRL Value:1 ECert:false) (Name:hf.Registrar.Attributes Value: ECert:false) (Name:hf.AffiliationMgr Value:1 ECert:fa
lse) (Name:hf.Registrar.Roles Value: ECert:false) (Name:hf.Registrar.DelegateRoles Value: ECert:false) (Name:hf.Revoker Value:1 ECert:false) (Name:hf.IntermediateCA Value:1 ECert:false)]
Name: peer0, Type: peer, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:peer0 ECert:true) (Name:hf.Type Value:peer ECert:true) (Name:hf.Affiliation Value: ECert:true)]
Name: user1, Type: client, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:user1 ECert:true) (Name:hf.Type Value:client ECert:true) (Name:hf.Affiliation Value: ECert:true)]
Name: orgladmin, Type: admin, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:orgladmin ECert:true) (Name:hf.Type Value:admin ECert:true) (Name:hf.Affiliation Value: ECert:tru
e)]
Name: SE1, Type: client, Affiliation: , Max Enrollments: 1, Attributes: [(Name:hf.EnrollmentID Value:SE1 ECert:true) (Name:hf.Type Value:client ECert:true) (Name:hf.Affiliation Value: ECert:true)]
vivekkukkapalli@Viveks-MacBook-Pro fabric-samples % bin/fabric-ca-client identity list
Name: admin, Type: client, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.GenCRL Value:1 ECert:false) (Name:hf.Registrar.Attributes Value: ECert:false) (Name:hf.AffiliationMgr Value:1 ECert:fa
lse) (Name:hf.Registrar.Roles Value: ECert:false) (Name:hf.Registrar.DelegateRoles Value: ECert:false) (Name:hf.Revoker Value:1 ECert:false) (Name:hf.IntermediateCA Value:1 ECert:false)]
Name: peer0, Type: peer, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:peer0 ECert:true) (Name:hf.Type Value:peer ECert:true) (Name:hf.Affiliation Value: ECert:true)]
Name: user1, Type: client, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:user1 ECert:true) (Name:hf.Type Value:client ECert:true) (Name:hf.Affiliation Value: ECert:true)]
Name: orgladmin, Type: admin, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:orgladmin ECert:true) (Name:hf.Type Value:admin ECert:true) (Name:hf.Affiliation Value: ECert:tru
e)]
Name: SE1, Type: peer, Affiliation: , Max Enrollments: 1, Attributes: [(Name:hf.EnrollmentID Value:SE1 ECert:true) (Name:hf.Type Value:peer ECert:true) (Name:hf.Affiliation Value: ECert:true)]
vivekkukkapalli@Viveks-MacBook-Pro fabric-samples % bin/fabric-ca-client identity list
Name: admin, Type: client, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.GenCRL Value:1 ECert:false) (Name:hf.Registrar.Attributes Value: ECert:false) (Name:hf.AffiliationMgr Value:1 ECert:fa
lse) (Name:hf.Registrar.Roles Value: ECert:false) (Name:hf.Registrar.DelegateRoles Value: ECert:false) (Name:hf.Revoker Value:1 ECert:false) (Name:hf.IntermediateCA Value:1 ECert:false)]
Name: peer0, Type: peer, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:peer0 ECert:true) (Name:hf.Type Value:peer ECert:true) (Name:hf.Affiliation Value: ECert:true)]
Name: user1, Type: client, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:user1 ECert:true) (Name:hf.Type Value:client ECert:true) (Name:hf.Affiliation Value: ECert:true)]
Name: orgladmin, Type: admin, Affiliation: , Max Enrollments: -1, Attributes: [(Name:hf.EnrollmentID Value:orgladmin ECert:true) (Name:hf.Type Value:admin ECert:true) (Name:hf.Affiliation Value: ECert:tru
e)]
vivekkukkapalli@Viveks-MacBook-Pro fabric-samples %
```

Attaching code snippets

1. Getting Fabric Users



```
const ccpPath = path.resolve(__dirname, '..', '..', 'test-network', 'organizations', 'peerOrganizations', 'org1.example.com');
const ccp = JSON.parse(fs.readFileSync(ccpPath, 'utf8'));

// Create a new CA client for interacting with the CA.
const caInfo = ccp.certificateAuthorities[0].payload.caInfo;
const caTLSCACerts = caInfo.tlsCACerts.pem;
const ca = new FabricCAServices(caInfo.url, { trustedRoots: caTLSCACerts, verify: false }, caInfo.caName);

// Create a new file system based wallet for managing identities.
const walletPath = path.join(process.cwd(), 'resPayload', 'wallet');
const wallet = await Wallets.newFileSystemWallet(walletPath);

const adminIdentity = await wallet.get(resPayload.adminIdentity);
if (!adminIdentity) {
  console.log('An identity for the admin user ' + resPayload.adminIdentity + ' does not exist in the wallet');
  console.log('Run the enrollAdmin.js application before retrying');
  return;
}

try {
  const provider = wallet.getProviderRegistry().getProvider(adminIdentity.type);
  const adminUser = await provider.getUserContext(adminIdentity, resPayload.adminIdentity);
  const identityService = ca.newIdentityService();

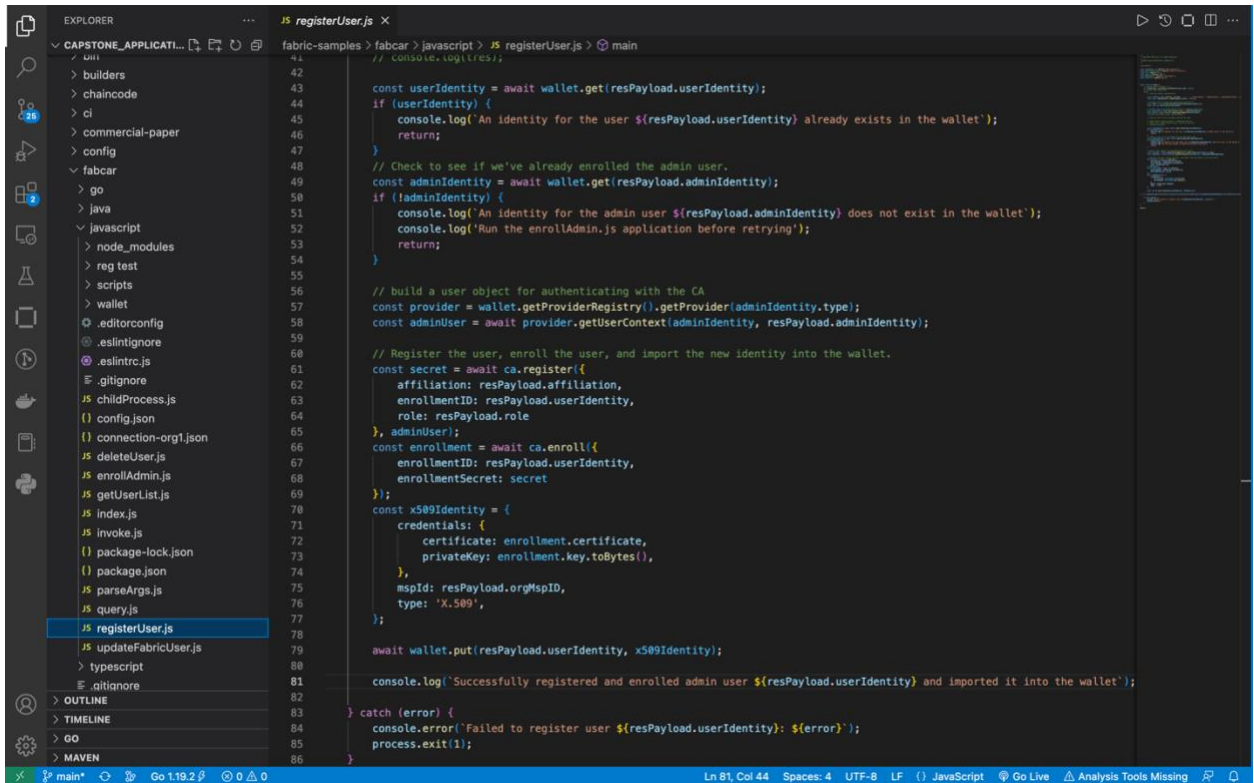
  const userList = await identityService.getAll(adminUser);

  console.log(JSON.stringify(userList));
} catch (error) {
  throw new Error(error);
}

} catch (error) {
  console.error('Failed to get list of users from the wallet: ' + error);
  process.exit(1);
}

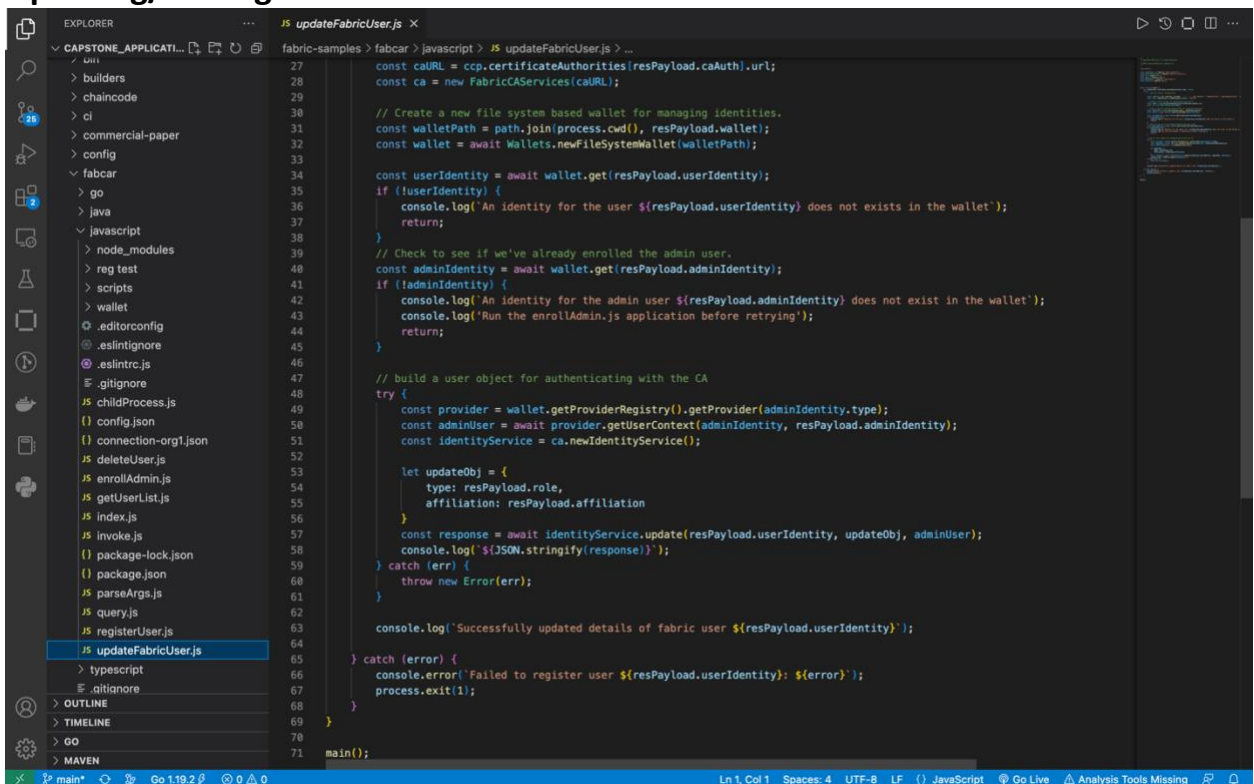
main();
```

2. Registering a new user



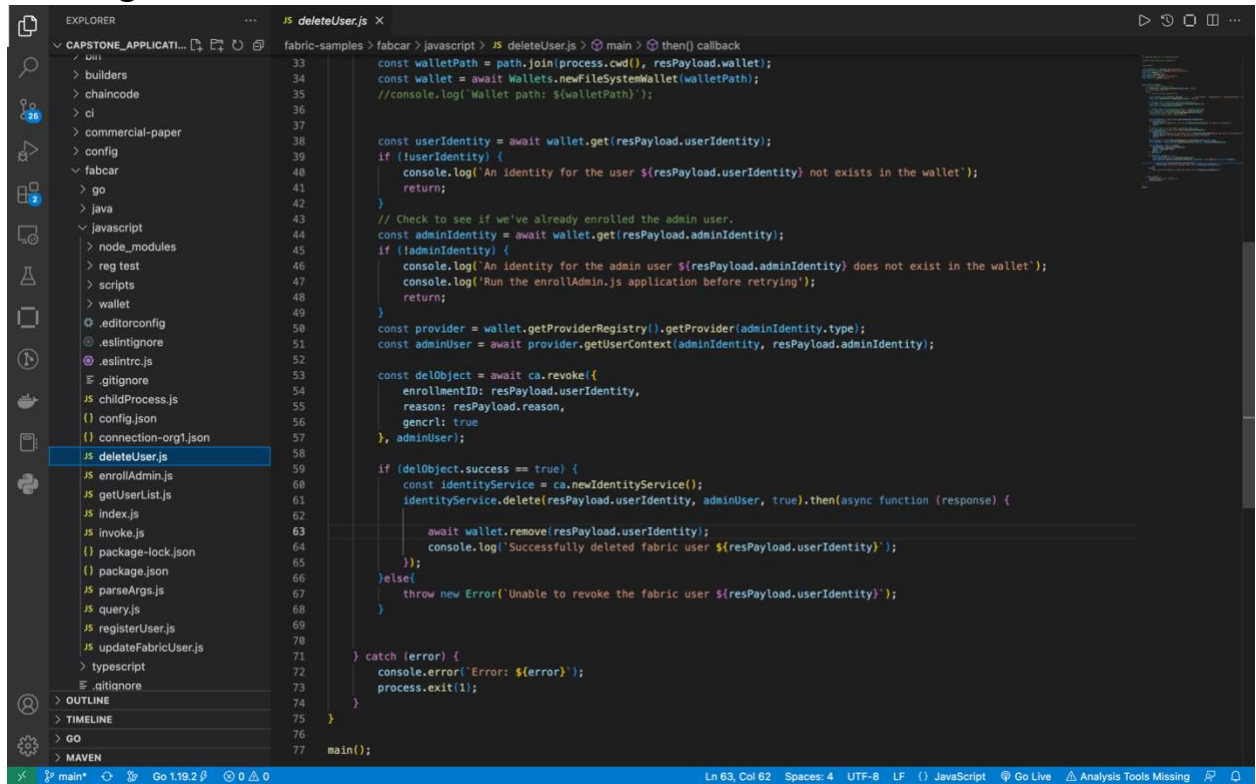
```
41 // console.log(res);
42
43 const userIdentity = await wallet.get(resPayload.userIdentity);
44 if (userIdentity) {
45   console.log('An identity for the user ${resPayload.userIdentity} already exists in the wallet');
46   return;
47 }
48
49 // Check to see if we've already enrolled the admin user.
50 const adminIdentity = await wallet.get(resPayload.adminIdentity);
51 if (!adminIdentity) {
52   console.log('An identity for the admin user ${resPayload.adminIdentity} does not exist in the wallet');
53   console.log('Run the enrollAdmin.js application before retrying');
54   return;
55 }
56
57 // build a user object for authenticating with the CA
58 const provider = wallet.getProviderRegistry().getProvider(adminIdentity.type);
59 const adminUser = await provider.getUserContext(adminIdentity, resPayload.adminIdentity);
60
61 // Register the user, enroll the user, and import the new identity into the wallet.
62 const secret = await ca.register({
63   affiliation: resPayload.affiliation,
64   enrollmentID: resPayload.userIdentity,
65   role: resPayload.role
66 }, adminUser);
67 const enrollment = await ca.enroll({
68   enrollmentID: resPayload.userIdentity,
69   enrollmentSecret: secret
70 });
71 const x509Identity = {
72   credentials: {
73     certificate: enrollment.certificate,
74     privateKey: enrollment.key.toBytes(),
75   },
76   mspid: resPayload.orgMspID,
77   type: 'X.509',
78 };
79
80 await wallet.put(resPayload.userIdentity, x509Identity);
81
82 console.log('Successfully registered and enrolled admin user ${resPayload.userIdentity} and imported it into the wallet');
83
84 } catch (error) {
85   console.error('Failed to register user ${resPayload.userIdentity}: ${error}');
86   process.exit(1);
87 }
```

3. Updating/editing user details



```
27 const caURL = ccp.certificateAuthorities[resPayload.caAuth.url];
28 const ca = new FabricCAServices(caURL);
29
30 // Create a new file system based wallet for managing identities.
31 const walletPath = path.join(process.cwd(), resPayload.wallet);
32 const wallet = await Wallets.newFileSystemWallet(walletPath);
33
34 const userIdentity = await wallet.get(resPayload.userIdentity);
35 if (!userIdentity) {
36   console.log('An identity for the user ${resPayload.userIdentity} does not exist in the wallet');
37   return;
38 }
39
40 // Check to see if we've already enrolled the admin user.
41 const adminIdentity = await wallet.get(resPayload.adminIdentity);
42 if (!adminIdentity) {
43   console.log('An identity for the admin user ${resPayload.adminIdentity} does not exist in the wallet');
44   console.log('Run the enrollAdmin.js application before retrying');
45   return;
46 }
47
48 // build a user object for authenticating with the CA
49 try {
50   const provider = wallet.getProviderRegistry().getProvider(adminIdentity.type);
51   const adminUser = await provider.getUserContext(adminIdentity, resPayload.adminIdentity);
52   const identityService = ca.newIdentityService();
53
54   let updateObj = {
55     type: resPayload.role,
56     affiliation: resPayload.affiliation
57   };
58   const response = await identityService.update(resPayload.userIdentity, updateObj, adminUser);
59   console.log(`${JSON.stringify(response)}`);
60 } catch (err) {
61   throw new Error(err);
62 }
63
64 console.log('Successfully updated details of fabric user ${resPayload.userIdentity}');
65
66 } catch (error) {
67   console.error('Failed to register user ${resPayload.userIdentity}: ${error}');
68   process.exit(1);
69 }
70
71 main();
```

4. Deleting the fabric user



The screenshot shows the Visual Studio Code editor with the `deleteUser.js` file open. The Explorer sidebar on the left shows the project structure, with `deleteUser.js` selected under the `javascript` folder. The main editor displays the following JavaScript code:

```
33 const walletPath = path.join(process.cwd(), resPayload.wallet);
34 const wallet = await Wallets.newFileSystemWallet(walletPath);
35 //console.log('Wallet path: ${walletPath}');
36
37
38 const userIdentity = await wallet.get(resPayload.userIdentity);
39 if (!userIdentity) {
40   console.log('An identity for the user ${resPayload.userIdentity} not exists in the wallet');
41   return;
42 }
43
44 // Check to see if we've already enrolled the admin user.
45 const adminIdentity = await wallet.get(resPayload.adminIdentity);
46 if (!adminIdentity) {
47   console.log('An identity for the admin user ${resPayload.adminIdentity} does not exist in the wallet');
48   console.log('Run the enrollAdmin.js application before retrying');
49   return;
50 }
51
52 const provider = wallet.getProviderRegistry().getProvider(adminIdentity.type);
53 const adminUser = await provider.getUserContext(adminIdentity, resPayload.adminIdentity);
54
55 const delObject = await ca.revoke({
56   enrollmentID: resPayload.userIdentity,
57   reason: resPayload.reason,
58   gencl: true
59 }, adminUser);
60
61 if (delObject.success == true) {
62   const identityService = ca.newIdentityService();
63   identityService.delete(resPayload.userIdentity, adminUser, true).then(async function (response) {
64     await wallet.remove(resPayload.userIdentity);
65     console.log('Successfully deleted fabric user ${resPayload.userIdentity}');
66   });
67 } else {
68   throw new Error('Unable to revoke the fabric user ${resPayload.userIdentity}');
69 }
70
71 } catch (error) {
72   console.error('Error: ${error}');
73   process.exit(1);
74 }
75
76
77 main();
```

The status bar at the bottom indicates the file is at line 63, column 62, with 4 spaces, UTF-8 encoding, and LF line endings. The language is set to JavaScript.