Exercise (Instructions): Loopback Data Sources and Access Control

Objectives and Outcomes

In this exercise, you will continue the exploration of Loopback. You will learn to set up a MongoDB as a data source and then set up access controls on the REST API endpoints. At the end of this exercise, you will be able to:

- Define data sources to be used by your Loopback server
- Set up access controls to various REST API end points.

Setting up a Data Source

- At the command prompt type the following to set up a MongoDB database as a data source:

```
slc loopback:datasource
```

- When you are prompted, enter the following as the information:

```
Data Source Name: MongoDB
Connector: Mongo DB connector
Host: localhost
Port: 27017
username & password: (empty)
Database Name: conFusion
```

- Say yes to installing the Loopback MongoDB connector.
- Open *model-config.json* file in the *server* subfolder of the *loopback-server* folder, and set the data source for*dishes*, *Role*, *RoleMapping* and ACL as MongoDB.

Implementing Access Control

- Add another model called *Customer* by typing the following at the prompt:

```
slc loopback:model
```

- Choose the following as the options:

```
Model Name: Customer
Data Source: MongoDB
Model's Base Class: User
REST API: Yes
```

No other properties need to be added. Just hit enter when prompted for property name.

- Now you will set up the access control list (ACL) to deny access to everyone for all the routes. Type at the command prompt:

```
slc loopback:acl
```

- Wnen prompted select the following:

```
(all existing models)
All methods and properties
All (match all types)
All users
Explicitly deny access
```

- Again set up the next ACL with the following options, to enable GET access to all authenticated users:

```
(all existing models)
All methods and properties
Read
Any authenticated users
Explicitly grant access
```

- The final ACL will be set up to allow only Admins to perform all operations:

```
dishes
A single method
create
Other users
role: admin
Explicitly grant access
```

- Now initialize the server with two user accounts, one of which is an admin, set up the following boot script in the*server/boot* folder in a file named *script.js*:

```
module.exports = function(app) {
var MongoDB = app.dataSources.MongoDB;

MongoDB.automigrate('Customer', function(err) {
   if (err) throw (err);
   var Customer = app.models.Customer;

   Customer.create([
    {username: 'Admin', email: 'admin@admin.com', password: 'abcdef'},
    {username: 'muppala', email: 'muppala@ust.hk', password: 'abcdef'}
  ], function(err, users) {
    if (err) return cb(err);
     var Role = app.models.Role;
    var RoleMapping = app.models.RoleMapping;

    //create the admin role
    Role.create({
      name: 'admin'
    }, function(err, role) {
      if (err) cb(err);
       //make admin
      role.principals.create({
        principalType: RoleMapping.USER,
        principalId: users[0].id
      }, function(err, principal) {
        if (err) throw (err);
      });
    });
  });
});

};
```

- Save the changes and start the server by typing "node ." at the prompt.
- Explore the server by logging in as Admin and the regular user.

Conclusions

In this exercise you learnt about setting up data sources and access control in a Loopback server.