Configuration Web Services

These web services will allow you to manage configuration settings in Yellowfin, as well as enter an admin mode.

Basic Configuration Functions

The following set of web services can be used to update system and customer configuration settings in Yellowfin.

This function updates the system configuration settings in Yellowfin. This works by changing the setting details directly in the Configuration table in Yellowfin's database, however only the general system settings are affected, not any custom settings. Yellowfin determines these by checking the **ConfigTypeCode** column for the value "SYSTEM". Therefore, this mainly effects the default organization.

Once the database table has been updated, you will need to restart Yellowfin for these changes to take effect.

Request Parameters

The following parameters should be passed with this request:

Request Element	Data Type	Description
LoginId	String	An admin account to connect to Yellowfin web services. This can be the user ID or the email address, depending on the Logon ID method. This account must have the "web services" role enabled, and must belong to the default (i.e. primary) org.
Password	String	Password of the above account.
OrgId	Integer	Default (i.e. primary) organization ID within Yellowfin. Always set this to 1.
Function	String	Web service function. Set this to "SAVECONFIGURATION".
Parameters	String[]	Array of strings. This parameter is used to pass the configuration settings. The first string should be the content for **configCode**, and the second is for **configData** of Configuration table.
Client	AdministrationClientOrg	This optional parameter is used to specify a particular client organization whose configurations are to be updated. However, if one is not specified, then the configuration settings will be applied to the default org. It must be noted, however, that the majority of the configuration settings are global and cannot be set up for a particular client.

Below are the mandatory parameters that you need to set in the **AdministrationClientOrg** object to create a new user:

Parameter	Data Type	Description
ClientReferenceId	String	The unique ID used to identify a client.

Request Example

Below is a SOAP XML example for this request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:web="http://webservices.web.
mi.hof.com/">
   <soapenv:Header/>
   <soapenv:Body>
      <web:remoteAdministrationCall>
         <arg0>
          <loginId>admin@yellowfin.com.au</loginId>
            <password>test</password>
            <orgId>1</orgId>
            <function>SAVECONFIGURATION</function>
            <parameters>Simple_Authentication</parameters>
            <parameters>true</parameters>
         </arg0>
      </web:remoteAdministrationCall>
   </soapenv:Body>
</soapenv:Envelope>
```

Response Parameters

The returned response will contain these parameters:

Response Element	Data Type	Description
StatusCode	String	Status of the web service call. Possible values include: • SUCCESS • FAILURE

Response Example

The service will return the below response, according to our SOAP example:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
   <S:Body>
      <ns2:remoteAdministrationCallResponse xmlns:ns2="http://webservices.web.mi.hof.com/">
         <return>
            <errorCode>0</errorCode>
            <messages>Successfully Authenticated User: admin@yellowfin.com.au</messages>
            <messages>Web Service Request Complete</messages>
            <sessionId>15359eb5b3d7d2c63c6f43affe5f3b45</sessionId>
            <statusCode>SUCCESS</statusCode>
         </return>
      </ns2:remoteAdministrationCallResponse>
   </S:Body>
</S:Envelope>
```

Instructions

See below for step-by-step instructions on how to perform this call, using a Java example:

- Define the request for this function, which includes logging in as the admin user and specifying the web service call to perform:

```
AdministrationServiceRequest rsr = new AdministrationServiceRequest();

rsr.setLoginId("admin@yellowfin.com.au");
rsr.setPassword("test");
rsr.setOrgId(1);
rsr.setFunction("SAVECONFIGURATION");
```

- Pass the configuration setting to be updated in the Parameters object. The code example below sets Yellowfin's authentication method to Simple.

```
rsr.setParameters(new String[]{"SIMPLE_AUTHENTICATION","TRUE"});
```

- Once the request is configured, perform the call:

```
AdministrationServiceResponse rs = adminService.remoteAdministrationCall(rsr);
```

Initialize the Administration web service. Click here to learn how to do this.

- Add the following code to retrieve the response. The response will contain the StatusCode. (See details in the Response Parameters table above.)

```
if ("SUCCESS".equals(rs.getStatusCode()) ) {
                    out.write("<br>Success");
                    }
                    else {
                    out.write("<br>Failure");
                    out.write(" Code: " + rs.getErrorCode());
                    }
```

Complete Example

Below is a full example of this web service call. To use it for yourself, carry out the following the steps:

1. Copy the code and save it as ws_saveconfiguration.jsp.
2. Put the file in the root folder: *Yellowfin/appserver/webapps/ROOT*.
3. Adjust the host, port, and admin user details according to your environment.
4. Run *http://<host>:<port>/ws_saveconfiguration.jsp* from your Internet browser.

```
<%
/*                  ws_SAVECONFIGURATION.jsp                                */
%>
<%@ page language="java" contentType="text/html; charset=UTF-8" %>
<%@ page import="com.hof.util.*, java.util.*, java.text.*" %>
<%@ page import="com.hof.web.form.*" %>
<%@ page import="com.hof.mi.web.service.*" %>
<%

/*
        This example sets Yellowfin authentication method to Simple.
    That means, once Yellowfin has been restarted, the
    LOGINUSERNOPASSWORD call can be used to log users into Yellowfin with no password provided.
*/

AdministrationServiceService s_adm = new AdministrationServiceServiceLocator("localhost",8080, "/services
/AdministrationService", false);           // adjust host and port number
AdministrationServiceSoapBindingStub adminService = (AdministrationServiceSoapBindingStub) s_adm.
getAdministrationService();
AdministrationServiceRequest rsr = new AdministrationServiceRequest();

rsr.setLoginId("admin@yellowfin.com.au");             // provide your Yellowfin webservices admin account
rsr.setPassword("test");                        // set to the password of the account above
rsr.setOrgId(1);
rsr.setFunction("SAVECONFIGURATION");

rsr.setParameters(new String[]{"SIMPLE_AUTHENTICATION","TRUE"});
AdministrationServiceResponse rs = adminService.remoteAdministrationCall(rsr);

if ("SUCCESS".equals(rs.getStatusCode()) ) {
                    out.write("<br>Success");
                    }
                    else {
                    out.write("<br>Failure");
                    out.write(" Code: " + rs.getErrorCode());
                    }
%>
```

This function updates the custom configuration settings in Yellowfin. The custom parameters are those that can be applied to a client organization. This works by changing the details directly in the Configuration table in Yellowfin's database. Yellowfin determines which settings to change by checking the **ConfigTypeCode** column for the value "CUSTOM".

Once the database table has been updated, you will need to restart Yellowfin for these changes to take affect.

Request Parameters

The following parameters should be passed with this request:

Request Element	Data Type	Description
LoginId	String	An admin account to connect to Yellowfin web services. This can be the user ID or the email address, depending on the Logon ID method. This account must have the "web services" role enabled, and must belong to the default (i.e. primary) org.
Password	String	Password of the above account.
OrgId	Integer	Default (i.e. primary) organization ID within Yellowfin. Always set this to 1.
Function	String	Web service function. Set this to " SAVECUSTOMPARAMETER".

Parameters	String[]	Array of strings. Set the first string to the value of **configCode**, and the second to that of the **configData** column of the Configuration table.
Client	AdministrationClientOrg	This optional parameter is used to specify a particular client organization whose configurations are to be updated. However, if one is not specified, then the configuration settings will be applied to the default org.

Below are the mandatory parameters that you need to set in the **AdministrationClientOrg** object to create a new user:

Parameter	Data Type	Description
ClientReferenceId	String	The unique ID used to identify a client organization.

Request Example

Below is a SOAP XML example for this request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:web="http://webservices.web.
mi.hof.com/">
   <soapenv:Header/>
   <soapenv:Body>
      <web:remoteAdministrationCall>
         <arg0>
          <loginId>admin@yellowfin.com.au</loginId>
            <password>test</password>
            <orgId>1</orgId>
            <function>SAVECUSTOMPARAMETER</function>
            <parameters>Simple_Authentication</parameters>
            <parameters>true</parameters>
         </arg0>
      </web:remoteAdministrationCall>
   </soapenv:Body>
</soapenv:Envelope>
```

Response Parameters

The returned response will contain these parameters:

Response Element	Data Type	Description
StatusCode	String	Status of the web service call. Possible values include: • SUCCESS • FAILURE

Response Example

The service will return the below response, according to our SOAP example:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
   <S:Body>
      <ns2:remoteAdministrationCallResponse xmlns:ns2="http://webservices.web.mi.hof.com/">
         <return>
            <errorCode>0</errorCode>
            <messages>Successfully Authenticated User: admin@yellowfin.com.au</messages>
            <messages>Web Service Request Complete</messages>
            <sessionId>471080457c1359cbf09c29cd348f97d1</sessionId>
            <statusCode>SUCCESS</statusCode>
         </return>
      </ns2:remoteAdministrationCallResponse>
   </S:Body>
</S:Envelope>
```

Instructions

See below for step-by-step instructions on how to perform this call, using a Java example:

- Define the request for this function, which includes logging in as the admin user and specifying the web service call to perform:

```
AdministrationServiceRequest rsr = new AdministrationServiceRequest();

rsr.setLoginId("admin@yellowfin.com.au");
rsr.setPassword("test");
rsr.setOrgId(1);
rsr.setFunction("SAVECUSTOMPARAMETER");
```

- Pass the configuration setting to be updated in the Parameters object. The code example below sets Yellowfin's authentication method to 'Simple'.

```
rsr.setParameters(new String[]{"SIMPLE_AUTHENTICATION","TRUE"});
```

- Once the request is configured, perform the call:

```
AdministrationServiceResponse rs = adminService.remoteAdministrationCall(rsr);
```

  Initialize the Administration web service. Click here to learn how to do this.

- Add the following code to retrieve the response. The response will contain the StatusCode. (See details in the Response Parameters table above.)

```
if ("SUCCESS".equals(rs.getStatusCode()) ) {
                     out.write("<br>Success");
                     }
                     else {
                     out.write("<br>Failure");
                     out.write(" Code: " + rs.getErrorCode());
                     }
```

Complete Example

Below is a full example of this web service call. To use it for yourself, carry out the following the steps:

1. Copy the code and save it as ws_savecustomparameter.jsp.

2. Put the file in the root folder: *Yellowfin/appserver/webapps/ROOT*.
3. Adjust the host, port, and admin user details according to your environment.
4. Run *http://<host>:<port>/ws_savecustomparameter.jsp* from your Internet browser.

```
<%
/*                  ws_savecustomparameter.jsp                      */
%>
<%@ page language="java" contentType="text/html; charset=UTF-8" %>
<%@ page import="com.hof.util.*, java.util.*, java.text.*" %>
<%@ page import="com.hof.web.form.*" %>
<%@ page import="com.hof.mi.web.service.*" %>
<%

/*
        This example set Yellowfin authentication method to Simple.
    That means, once Yellowfin has been restarted,
    LOGINUSERNOPASSWORD call can be used to log users into Yellowfin with no password provided.
*/

AdministrationServiceService s_adm = new AdministrationServiceServiceLocator("localhost",8080, "/services
/AdministrationService", false);    // adjust host and port number
AdministrationServiceSoapBindingStub adminService = (AdministrationServiceSoapBindingStub) s_adm.
getAdministrationService();
AdministrationServiceRequest rsr = new AdministrationServiceRequest();

rsr.setLoginId("admin@yellowfin.com.au");             // provide your Yellowfin web services admin account
rsr.setPassword("test");                        // change to the password of the account above
rsr.setOrgId(1);
rsr.setFunction("SAVECUSTOMPARAMETER");

rsr.setParameters(new String[]{"SIMPLE_AUTHENTICATION","TRUE"});
AdministrationServiceResponse rs = adminService.remoteAdministrationCall(rsr);

if ("SUCCESS".equals(rs.getStatusCode()) ) {
                    out.write("<br>Success");
                    }
                    else {
                    out.write("<br>Failure");
                    out.write(" Code: " + rs.getErrorCode());
                    }
%>
```

This web service retrieves the current system configuration details of Yellowfin. This is done by specifying a **ConfigCode** element; the system then fetches the values saved in that element's corresponding **ConfigData** column of the Configuration table in Yellowfin's database. Note that only system (and not custom) parameters are loaded, as Yellowfin checks for values that contain "SYSTEM" in the **ConfigTypeCode** column.

For instance, using this you can verify if simple authentication is enabled, that allows users to log in without a password.

Request Parameters

The following parameters should be passed with this request:

Request Element	Data Type	Description
LoginId	String	An admin account to connect to Yellowfin web services. This can be the user ID or the email address, depending on the Logon ID method. This account must have the "web services" role enabled, and must belong to the default (i.e. primary) org.
Password	String	Password of the above account.

OrgId	Integer	Default (i.e. primary) organization ID within Yellowfin. Always set this to 1.
Function	String	Web service function. Set this to "LOADCONFIGURATION".
Parameters	String[]	Array of strings. This is used to pass the value of the ConfigCode element.
Client	AdministrationClientOrg	This optional parameter is used to specify a particular client organization whose configurations are to be updated. However, if one is not specified, then the configuration settings will be applied to the default org.

Below are the mandatory parameters that you need to set in the AdministrationClientOrg object to create a new user:

Parameter	Data Type	Description
ClientReferenceId	String	The unique ID used to identify a client organization.

Response Parameters

The returned response will contain these parameters:

Response Element	Data Type	Description
StatusCode	String	Status of the web service call. Possible values include: • SUCCESS • FAILURE
BinaryData	String	Contains the configData value of the specified Configuration.configCode element (from Yellowfin's Configuration database). Null will be returned if no such ConfigCode is found in the Configuration table.

Instructions

See below for step-by-step instructions on how to perform this call, using a Java example:

- Define the request for this function, which includes logging in as the admin user and specifying the web service call to perform:

```
AdministrationServiceRequest rsr = new AdministrationServiceRequest();

rsr.setLoginId("admin@yellowfin.com.au");
rsr.setPassword("test");
rsr.setOrgId(1);
rsr.setFunction("LOADCONFIGURATION");
```

- Pass the configuration setting to be updated in the Parameters object. The code example below sets Yellowfin's authentication method to 'Simple'.

```
rsr.setParameters(new String[]{"SIMPLE_AUTHENTICATION"});
```

- Once the request is configured, perform the call:

```
AdministrationServiceResponse rs = adminService.remoteAdministrationCall(rsr);
```

Initialize the Administration web service. Click here to learn how to do this.

- Add the following code to retrieve the response. (See the response that is returned in the Response Parameters table above.)

```
if ("SUCCESS".equals(rs.getStatusCode()) ) {
                      out.write("<br>Success");
                                String data = rs.getBinaryData();
                      out.write("<br>ConfigData: " + data);
                      }
else {
                      out.write("<br>Failure");
                      out.write(" Code: " + rs.getErrorCode());
}
```

Complete Example

Below is a full example of this web service call. To use it for yourself, carry out the following the steps:

1. Copy the code and save it as ws_loadconfiguration.jsp.
2. Put the file in the root folder: *Yellowfin/appserver/webapps/ROOT*.
3. Adjust the host, port, and admin user details according to your environment.
4. Run *http://<host>:<port>/ws_loadconfiguration.jsp* from your Internet browser.

```
<%
/*                  ws_loadconfiguration.jsp                          */
%>
<%@ page language="java" contentType="text/html; charset=UTF-8" %>
<%@ page import="com.hof.util.*, java.util.*, java.text.*" %>
<%@ page import="com.hof.web.form.*" %>
<%@ page import="com.hof.mi.web.service.*" %>
<%
AdministrationServiceService s_adm = new AdministrationServiceServiceLocator("localhost",8080, "/services
/AdministrationService", false);            // adjust host and port number
AdministrationServiceSoapBindingStub adminService = (AdministrationServiceSoapBindingStub) s_adm.
getAdministrationService();
AdministrationServiceRequest rsr = new AdministrationServiceRequest();

rsr.setLoginId("admin@yellowfin.com.au");              // provide your Yellowfin webservices admin account
rsr.setPassword("test");                                                    // change to be the
password of the account above
rsr.setOrgId(1);
rsr.setFunction("LOADCONFIGURATION");

rsr.setParameters(new String[]{"SIMPLE_AUTHENTICATION"});
AdministrationServiceResponse rs = adminService.remoteAdministrationCall(rsr);

if ("SUCCESS".equals(rs.getStatusCode()) ) {
                     out.write("<br>Success");
String data = rs.getBinaryData();
                     out.write("<br>ConfigData: " + data);
                     }
else {
                     out.write("<br>Failure");
                     out.write(" Code: " + rs.getErrorCode());
}
%>
```

This web service retrieves the custom configuration details of Yellowfin. This is done by specifying a ConfigCode element; the system then fetches the values saved in that element's corresponding ConfigData column of the Configuration table in Yellowfin's database. Only custom parameters are loaded, as Yellowfin checks for values that contain "CUSTOM" in the ConfigTypeCode column.

For instance, using this you can verify if simple authentication is enabled, that allows users to log in without a password.

Request Parameters

The following parameters should be passed with this request:

Request Element	Data Type	Description
LoginId	String	An admin account to connect to Yellowfin web services. This can be the user ID or the email address, depending on the Logon ID method. This account must have the “web services” role enabled, and must belong to the default (i.e. primary) org.
Password	String	Password of the above account.
OrgId	Integer	Default (i.e. primary) organization ID within Yellowfin. Always set this to 1.
Function	String	Web service function. Set this to " LOADCUSTOMPARAMETER".
Parameters	String[]	Array of strings. This is used to pass the value of the ConfigCode element.
Client	AdministrationClientOrg	This optional parameter is used to specify a particular client organization whose configurations are to be updated. However, if one is not specified, then the configuration settings will be applied to the default org.

Below are the mandatory parameters that you need to set in the **AdministrationClientOrg** object to create a new user:

Parameter	Data Type	Description
ClientReferenceId	String	The unique ID used to identify a client organization.

Response Parameters

The returned response will contain these parameters:

Response Element	Data Type	Description
StatusCode	String	Status of the web service call. Possible values include: • SUCCESS • FAILURE
BinaryData	String	Contains the configData value of the specified Configuration.configCode element (from Yellowfin's Configuration database). Null will be returned if no such ConfigCode is found in the Configuration table.

Instructions

See below for step-by-step instructions on how to perform this call, using a Java example:

- Define the request for this function, which includes logging in as the admin user and specifying the web service call to perform:

```
AdministrationServiceRequest rsr = new AdministrationServiceRequest();

rsr.setLoginId("admin@yellowfin.com.au");
rsr.setPassword("test");
rsr.setOrgId(1);
rsr.setFunction("LOADCONFIGURATION");
```

- Pass the configuration setting to be updated in the Parameters object. The code example below sets Yellowfin's authentication method to 'Simple'.

```
rsr.setParameters(new String[]{"SIMPLE_AUTHENTICATION"});
```

- Once the request is configured, perform the call:

```
AdministrationServiceResponse rs = adminService.remoteAdministrationCall(rsr);
```

Initialize the Administration web service. Click here to learn how to do this.

- Add the following code to retrieve the response. (See the response that is returned in the Response Parameters table above.)

```
if ("SUCCESS".equals(rs.getStatusCode()) ) {
                      out.write("<br>Success");
                                String data = rs.getBinaryData();
                      out.write("<br>ConfigData: " + data);
                      }
else {
                      out.write("<br>Failure");
                      out.write(" Code: " + rs.getErrorCode());
}
```

Complete Example

Below is a full example of this web service call. To use it for yourself, carry out the following the steps:

1. Copy the code and save it as ws_loadcustomparameter.jsp.
2. Put the file in the root folder: *Yellowfin/appserver/webapps/ROOT*.
3. Adjust the host, port, and admin user details according to your environment.
4. Run *http://<host>:<port>/ws_loadcustomparameter.jsp* from your Internet browser.

```
/*                 ws_loadcustomparameter.jsp                              */
%>
<%@ page language="java" contentType="text/html; charset=UTF-8" %>
<%@ page import="com.hof.util.*, java.util.*, java.text.*" %>
<%@ page import="com.hof.web.form.*" %>
<%@ page import="com.hof.mi.web.service.*" %>

<%
AdministrationServiceService s_adm = new AdministrationServiceServiceLocator("localhost",8080, "/services
/AdministrationService", false);           // adjust host and port number
AdministrationServiceSoapBindingStub adminService = (AdministrationServiceSoapBindingStub) s_adm.
getAdministrationService();
AdministrationServiceRequest rsr = new AdministrationServiceRequest();

rsr.setLoginId("admin@yellowfin.com.au");             // provide your Yellowfin web services admin account
rsr.setPassword("test");                       // set to the password of the account above
rsr.setOrgId(1);
rsr.setFunction("LOADCUSTOMPARAMETER");

rsr.setParameters(new String[]{"SIMPLE_AUTHENTICATION"});
AdministrationServiceResponse rs = adminService.remoteAdministrationCall(rsr);

if ("SUCCESS".equals(rs.getStatusCode()) ) {
                    out.write("<br>Success");
String data = rs.getBinaryData();
                    out.write("<br>ConfigData: " + data);
                    }
else {
                    out.write("<br>Failure");
                    out.write(" Code: " + rs.getErrorCode());
}
%>
```

Admin Mode Functions

The below web services allow users to enter or leave the “Admin” mode. The admin mode can be set up with a specific set of user role permissions, which will be applied to all the users when the mode is enabled.

The web services below provide more detail on what happens when this mode is enabled or disabled.

This web service enables the Admin mode on a Yellowfin instance. You can use this service to pass a set of user permissions to be applied to all user roles. Doing so will change every user’s permissions to the one defined in this function, including currently active users, and those who log in after calling this function.

This means that if a user has only the basic “report consumer” role with read-only access, you can use this function to change their active session permissions and grant them full report access, including deleting reports. Similarly, you can use this web service to make Yellowfin non-editable, by removing the report writing functionality from all active users’ roles.

To set up these role functions, you’ll need to specify the function code and access level. Any functions not in the list will use the existing access levels for that role. Specifying no access level is the same as disabling a function.

In case of a cluster setup, this function can be set to notify specific nodes in the cluster to enable the admin mode, by using the Parameters request element.

Once this mode is entered, all new sessions will apply the permissions set in the mode. To return to the original user permissions for all active sessions, as well as future logins, simply disable this mode by passing the DISABLEADMINMODE.

If you already have the admin mode enabled, and want to set new access levels, then you will need to first disable the mode, and then enable it again with the updated access levels, since this mode can only be enabled if it is not already active.

Request Parameters

The following parameters should be passed with this request:

Request Element	Data Type	Description
LoginId	String	An admin account to connect to Yellowfin web services. This can be the user ID or the email address, depending on the Logon ID method. This account must have the “web services” role enabled, and must belong to the default (i.e. primary) org.
Password	String	Password of the above account.
OrgId	Integer	Default (i.e. primary) organization ID within Yellowfin. Always set this to 1.
Function	String	Web service function. Set this to " ENABLEADMINMODE".
RoleFunctionList	Map<String, String>	A list of role function names mapped to their access levels. These will be the role functions enabled in the admin mode. For each function code, provide an access level, using the CRUD notion. If the access level is not provided, the function will not be included. See the AdministrationFunction object definition for possible values.
Parameters	String[]	This is used in case of a cluster set up. This array is used to pass either a “true” or “TRUE” String to signal all nodes in the cluster to enable the admin mode. If no value is passed here, it will be set to false by default, thereby not enabling the admin mode.

Request Example

The following SOAP example shows the parameters that you can pass to this call:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:web="http://webservices.web.
mi.hof.com/">
   <soapenv:Header/>
   <soapenv:Body>
      <web:remoteAdministrationCall>
         <arg0>
            <loginId>admin@yellowfin.com.au</loginId>
            <password>test</password>
            <orgId>1</orgId>
            <orgRef>org1</orgRef>
            <function>ENABLEADMINMODE</function>
            <parameters>true</parameters>
            <roleFunctionList>
                    <key>MIREPORT</key>
                    <value>CRUD</value>
            </roleFunctionList>
         </arg0>
      </web:remoteAdministrationCall>
   </soapenv:Body>
</soapenv:Envelope>
```

Response Parameters

The returned response will contain these parameters:

Response Element	Data Type	Description
StatusCode	String	Status of the web service call. Possible values include: • SUCCESS • FAILURE

Response Elements

The service will return the below response, according to our SOAP example:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
   <S:Body>
      <ns2:remoteAdministrationCallResponse xmlns:ns2="http://webservices.web.mi.hof.com/">
         <return>
            <errorCode>0</errorCode>
            <messages>Successfully Authenticated User: admin@yellowfin.com.au</messages>
            <messages>Web Service Request Complete</messages>
            <sessionId>517ce15c32a079dc1528424c03d86691</sessionId>
            <statusCode>SUCCESS</statusCode>
         </return>
      </ns2:remoteAdministrationCallResponse>
   </S:Body>
</S:Envelope>
```

Instructions

See below for step-by-step instructions on how to perform this call, using a Java example:

- Define the request for this function, which includes logging in as the admin user and specifying the web service call to perform:

```
AdministrationServiceRequest asr = new AdministrationServiceRequest();


asr.setLoginId("admin@yellowfin.com.au");
asr.setPassword("test");
asr.setOrgId(new Integer(1));
asr.setFunction("ENABLEADMINMODE");
```

- In case of a cluster set up, alert all the other nodes in the cluster to enable the admin mode:

```
asr.setParameters(new String[] { "true" });
```

- Set the required role functions for the admin role. For each role function name, specify its access level:

```
Map<String, String> roleFunctionList = new HashMap<>();

roleFunctionList.put("SYSTEMINFO", "");
roleFunctionList.put("REPORTDASHBOARD", "R");
roleFunctionList.put("DASHPUBLIC", "R");
roleFunctionList.put("MIREPORT", "R");
roleFunctionList.put("STORYBOARD", "R");
```

- Add the list to the request:

```
asr.setRoleFunctionList(roleFunctionList);
```

- Once the request is configured, perform the call:

```
AdministrationServiceResponse rs = rssbs.remoteAdministrationCall(asr);
```

Initialize the Administration web service. Click here to learn how to do this.

- The response returned will contain the StatusCode parameter. (See the response that is returned in the Response Parameters table above.)

Complete Example

Below is a full example of this web service call. To use it for yourself, carry out the following the steps:

1. Copy the code and save it as ws_enableadminmode.jsp.
2. Put the file in the root folder: *Yellowfin/appserver/webapps/ROOT*.
3. Adjust the host, port, and admin user details according to your environment.
4. Run *http://<host>:<port>/ws_enableadminmode.jsp* from your Internet browser.

```
<%@ page language="java" contentType="text/html; charset=UTF-8" %>
<%@ page import="com.hof.util.*, java.util.*, java.text.*" %>
<%@ page import="com.hof.web.form.*" %>
<%@ page import="com.hof.mi.web.service.*" %>
<%
/*
Enable Admin Mode
Enables administrative mode with the list of functions in roleFunctionList.
A parameter is set to indicate this should notify the cluster.
*/

AdministrationServiceResponse rs = null;
AdministrationServiceRequest asr = new AdministrationServiceRequest();
AdministrationServiceService ts = new AdministrationServiceServiceLocator("localhost", 8080,
"/services/AdministrationService", false);
AdministrationServiceSoapBindingStub rssbs = (AdministrationServiceSoapBindingStub) ts
.getAdministrationService();

asr.setLoginId("admin@yellowfin.com.au");
asr.setPassword("test");
asr.setOrgId(new Integer(1));
asr.setFunction("ENABLEADMINMODE");
asr.setParameters(new String[] { "true" });

Map<String, String> roleFunctionList = new HashMap<>();
roleFunctionList.put("SYSTEMINFO", "");
roleFunctionList.put("REPORTDASHBOARD", "R");
roleFunctionList.put("DASHPUBLIC", "R");
roleFunctionList.put("MIREPORT", "R");
roleFunctionList.put("STORYBOARD", "R");

asr.setRoleFunctionList(roleFunctionList);

rs = rssbs.remoteAdministrationCall(asr);

if ("SUCCESS".equals(rs.getStatusCode())) {
for (String message : rs.getMessages()) {
out.write(message + "<BR>");
}
} else {
out.write("Failure");
out.write(" Code: " + rs.getErrorCode());
}
%>
```

This web service disables the admin mode on a Yellowfin instance. Doing so resets the user permissions for active users and future logins, back to the original role permissions. In case of a cluster environment, you will need to use a Parameters request element to pass a “true” or “True” string to notify all the nodes to disable the admin mode. If no value is specified in the element, then by “false/False” will be set by default, and the admin mode will not be disabled.

Request Parameters

The following parameters should be passed with this request:

Request Element	Data Type	Description
LoginId	String	An admin account to connect to Yellowfin web services. This can be the user ID or the email address, depending on the Logon ID method. This account must have the “web services” role enabled, and must belong to the default (i.e. primary) org.
Password	String	Password of the above account.
OrgId	Integer	Default (i.e. primary) organization ID within Yellowfin. Always set this to 1.
Function	String	Web service function. Set this to "DISABLEADMINMODE".
Parameters	String[]	This is used in case of a cluster set up. This array is used to pass either a “true” or “TRUE” string to signal all nodes in the cluster to disable the admin mode. If no value is passed here, it will be set to false by default, thereby not disabling the admin mode.

Request Example

The following SOAP example shows the parameters that you can pass to this call:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:web="http://webservices.web.
mi.hof.com/">
   <soapenv:Header/>
   <soapenv:Body>
      <web:remoteAdministrationCall>
         <arg0>
            <loginId>admin@yellowfin.com.au</loginId>
            <password>test</password>
            <orgId>1</orgId>
            <orgRef>org1</orgRef>
            <function>DISABLEADMINMODE</function>
            <parameters>true</parameters>
         </arg0>
      </web:remoteAdministrationCall>
   </soapenv:Body>
</soapenv:Envelope>
```

Response Parameters

The returned response will contain these parameters:

Response Element	Data Type	Description
StatusCode	String	Status of the web service call. Possible values include: • SUCCESS • FAILURE

Response Elements

The service will return the below response, according to our SOAP example:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
   <S:Body>
      <ns2:remoteAdministrationCallResponse xmlns:ns2="http://webservices.web.mi.hof.com/">
         <return>
            <errorCode>0</errorCode>
            <messages>Successfully Authenticated User: admin@yellowfin.com.au</messages>
            <messages>Web Service Request Complete</messages>
            <sessionId>b2aedaafa1350f99c904b88c5daa14e0</sessionId>
            <statusCode>SUCCESS</statusCode>
         </return>
      </ns2:remoteAdministrationCallResponse>
   </S:Body>
</S:Envelope>
```

Instructions

See below for step-by-step instructions on how to perform this call, using a Java example:

- Define the request for this function, which includes logging in as the admin user and specifying the web service call to perform:

```
AdministrationServiceRequest asr = new AdministrationServiceRequest();


asr.setLoginId("admin@yellowfin.com.au");
asr.setPassword("test");
asr.setOrgId(new Integer(1));
asr.setFunction("DISABLEADMINMODE");
```

- In case of a cluster set up, alert all the other nodes in the cluster to disable the admin mode:

```
asr.setParameters(new String[] { "true" });
```

- Once the request is configured, perform the call:

```
AdministrationServiceResponse rs = rssbs.remoteAdministrationCall(asr);
```

  Initialize the Administration web service. Click here to learn how to do this.

- The response returned will contain the StatusCode parameter. (See the response that is returned in the Response Parameters table above.)

Complete Example

Below is a full example of this web service call. To use it for yourself, carry out the following the steps:

1. Copy the code and save it as ws_disableadminmode.jsp.
2. Put the file in the root folder: *Yellowfin/appserver/webapps/ROOT*.
3. Adjust the host, port, and admin user details according to your environment.
4. Run *http://<host>:<port>/ws_disableadminmode.jsp* from your Internet browser.

```
<%@ page language="java" contentType="text/html; charset=UTF-8" %>
<%@ page import="com.hof.util.*, java.util.*, java.text.*" %>
<%@ page import="com.hof.web.form.*" %>
<%@ page import="com.hof.mi.web.service.*" %>
<%
/*
Disable Admin Mode
Disables administrative mode. A parameter is set to indicate this should notify the cluster.
*/
AdministrationServiceResponse rs = null;
AdministrationServiceRequest asr = new AdministrationServiceRequest();
AdministrationServiceService ts = new AdministrationServiceServiceLocator("localhost", 8080,
"/services/AdministrationService", false);
AdministrationServiceSoapBindingStub rssbs = (AdministrationServiceSoapBindingStub) ts
.getAdministrationService();
asr.setLoginId("admin@yellowfin.com.au");
asr.setPassword("test");
asr.setOrgId(new Integer(1));
asr.setFunction("DISABLEADMINMODE");
asr.setParameters(new String[] { "true" });
rs = rssbs.remoteAdministrationCall(asr);
if ("SUCCESS".equals(rs.getStatusCode())) {
for (String message : rs.getMessages()) {
out.write(message + "<BR>");
}
} else {
out.write("Failure");
out.write(" Code: " + rs.getErrorCode());
}
%>
```

This web service will retrieve all of the role functions, and some of their descriptive information, available in Yellowfin. The result is a 2D array of strings representing rows returned from a database query. The columns are:

- **Function Code:** The string used to define the function in the database.
- **Function Name:** The name of the function as seen in the role administration screen.
- **Function Description:**The description of the function as seen in the role administration screen.
- **Uses CRUD:** A flag indicating whether this function uses CRUD (Create, Read, Update, Delete) permissions.

Request Parameters

The following parameters should be passed with this request:

Request Element	Data Type	Description
LoginId	String	An admin account to connect to Yellowfin web services. This can be the user ID or the email address, depending on the Logon ID method. This account must have the “web services” role enabled, and must belong to the default (i.e. primary) org.
Password	String	Password of the above account.
OrgId	Integer	Default (i.e. primary) organization ID within Yellowfin. Always set this to 1.
Function	String	Web service function. Set this to "GETROLEFUNCTIONS".
Parameters	String[]	The parameters array is used to pass the password of your database, if one is set.

Request Example

The following SOAP example shows the parameters that you can pass to this call:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:web="http://webservices.web.
mi.hof.com/">
   <soapenv:Header/>
   <soapenv:Body>
      <web:remoteAdministrationCall>
         <arg0>
            <loginId>admin@yellowfin.com.au</loginId>
            <password>test</password>
            <orgId>1</orgId>
            <orgRef>org1</orgRef>
            <function>GETROLEFUNCTIONS</function>
         </arg0>
      </web:remoteAdministrationCall>
   </soapenv:Body>
</soapenv:Envelope>
```

Response Parameters

The returned response will contain these parameters:

Response Element	Data Type	Description
StatusCode	String	Status of the web service call. Possible values include: • SUCCESS • FAILURE
QueryResults	ReportRow[]	Object containing details of the role functions. See table below.

The following parameters will be returned in the ReportRow object:

ReportRow Element	Data Type	Description
Function Code	String	The string used to define the function in the database.
Function Name	String	The name of the function as seen in the role administration screen.
Function Description	String	The description of the function as seen in the role administration screen.
Uses CRUD	String	A flag indicating whether this function uses CRUD (Create, Read, Update, Delete) permissions.

Response Elements

The service will return the below response, according to our SOAP example:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
     <ns2:remoteAdministrationCallResponse xmlns:ns2="http://webservices.web.mi.hof.com/">
        <return>
           <errorCode>0</errorCode>
           <messages>Successfully Authenticated User: admin@yellowfin.com.au</messages>
           <messages>Web Service Request Complete</messages>
           <queryResults>
              <dataValue>ACCESSFILTER</dataValue>
              <dataValue>Access Filter</dataValue>
              <dataValue>Allows users to set or change the access filter on a report</dataValue>
              <dataValue>0</dataValue>
           </queryResults>
           <queryResults>
              <dataValue>CONTENTACCESS</dataValue>
              <dataValue>Folder Access</dataValue>
              <dataValue>Allow users to configure content folders.</dataValue>
              <dataValue>0</dataValue>
           </queryResults>
           .
                       .
                       .
           <queryResults>
              <dataValue>SHOWHEADERFOOTERADMIN</dataValue>
              <dataValue>Header/Footer Admin</dataValue>
              <dataValue>Allow the user to edit the Header and Footer displayed throughout the system.<
/dataValue>
              <dataValue>0</dataValue>
           </queryResults>
           <sessionId>31e386b7f19b97d6dcdea223f6b2f737</sessionId>
           <statusCode>SUCCESS</statusCode>
        </return>
     </ns2:remoteAdministrationCallResponse>
  </S:Body>
</S:Envelope>
```

Instructions

See below for step-by-step instructions on how to perform this call, using a Java example:

- Define the request for this function, which includes logging in as the admin user and specifying the web service call to perform:

```
AdministrationServiceRequest asr = new AdministrationServiceRequest();

asr.setLoginId("admin@yellowfin.com.au");
asr.setPassword("test");
asr.setOrgId(new Integer(1));
asr.setFunction("GETROLEFUNCTIONS");
```

- Provide access to your database by providing its password:

```
asr.setParameters(new String[] { "root" });
```

- Once the request is configured, perform the call:

```
AdministrationServiceResponse rs = rssbs.remoteAdministrationCall(asr);
```

  Initialize the Administration web service. Click here to learn how to do this.

- The response will contain the StatusCode, along with other information. See the Response Parameter table above for details on these. Save all the regular messages from the response.

```
for (String message : rs.getMessages()) {
out.write(message + "<BR>");
}
```

- Also create an HTML table with the role function details. (The column names will appear in the first row and then each row from the database query will be printed out after that.)

```
out.write("<table>");
out.write("<th>Function Code</th>");
out.write("<th>Function Name</th>");
out.write("<th>Function Description</th>");
out.write("<th>Uses CRUD</th>");
for (ReportRow row : rs.getQueryResults()) {
        out.write("<tr>");
        for (String data : row.getDataValue()) {
                out.write("<th>" + data + "</th>");
        }
        out.write("</tr>");
}
out.write("</table>");
```

Complete Example

Below is a full example of this web service call. To use it for yourself, carry out the following the steps:

1. Copy the code and save it as ws_getrolefunctions.jsp.
2. Put the file in the root folder: *Yellowfin/appserver/webapps/ROOT*.
3. Adjust the host, port, and admin user details according to your environment.
4. Run *http://<host>:<port>/ws_getrolefunctions.jsp* from your Internet browser.

```
<%@ page language="java" contentType="text/html; charset=UTF-8" %>
<%@ page import="com.hof.util.*, java.util.*, java.text.*" %>
<%@ page import="com.hof.web.form.*" %>
<%@ page import="com.hof.mi.web.service.*" %>
<%
/*
Get Role Functions                        ws_getrolefunctions.jsp
Returns all of the role functions available from the OrgFunction table with descriptions from the
OrgReferenceCodeDesc table. Also includes whether the function uses CRUD level access.
*/
AdministrationServiceResponse rs = null;
AdministrationServiceRequest asr = new AdministrationServiceRequest();
AdministrationServiceService ts = new AdministrationServiceServiceLocator("localhost", 8080, "/services
/AdministrationService", false);
AdministrationServiceSoapBindingStub rssbs = (AdministrationServiceSoapBindingStub) ts
.getAdministrationService();
asr.setLoginId("admin@yellowfin.com.au");
asr.setPassword("test");
asr.setOrgId(new Integer(1));
asr.setFunction("GETROLEFUNCTIONS");
asr.setParameters(new String[] { "root" }); // Database password

rs = rssbs.remoteAdministrationCall(asr);
if ("SUCCESS".equals(rs.getStatusCode())) {
for (String message : rs.getMessages()) {
out.write(message + "<BR>");
}
// Create a simple table with the results
out.write("<table>");
out.write("<th>Function Code</th>");
out.write("<th>Function Name</th>");
out.write("<th>Function Description</th>");
out.write("<th>Uses CRUD</th>");
for (ReportRow row : rs.getQueryResults()) {
out.write("<tr>");
for (String data : row.getDataValue()) {
out.write("<th>" + data + "</th>");
}
out.write("</tr>");
}
out.write("</table>");
} else {
out.write("Failure");
out.write(" Code: " + rs.getErrorCode());
}
%>
```