Introducing Cizer.Net Reporting Web Services!

In expanding our Cizer.Net reporting toolset, we have released a web service with the new version of our reporting software, version 4.0. This web service will allow other software products and services to communicate and interface with Cizer.Net in order to provide flexible solutions for any application that can make use of Cizer.Net embedded functionality which is based on Microsoft SQL Server Reporting Services.

For example, it provides the features that Application Service Providers and Software Publishers have been requesting since the introduction of Microsoft Reporting Services, for pre-packaged customizable web-based reporting components.

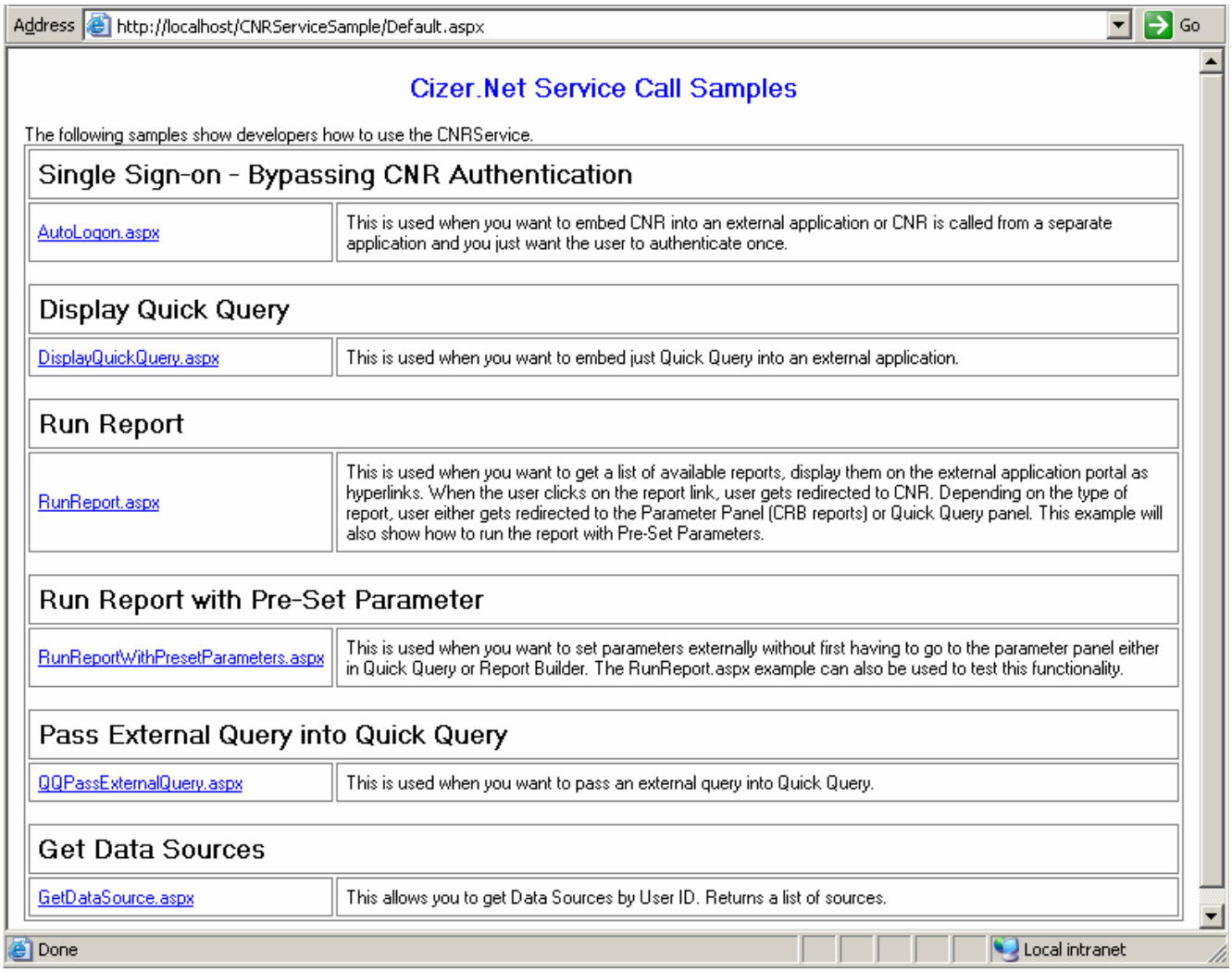
To help understand the functionality of this new Cizer.Net Web Service, a pre-built application has been developed that incorporates the various aspects of the web service. It should provide developers with clear examples of how to use it and how to implement it in some of their own solutions.

This document will act as a guide through the various functions of the web service.

Contents:

Main CNR Web Service - Sample Application

The main page of this web application is the **Default.aspx** page (shown below). It incorporates all of the functionality of the web service condensed into five types of examples.



Initial Setup Code

Before calling any of the functions in the web service, an instance of CNR Service must be set up and a user must be authenticated. Below is the example C# code:

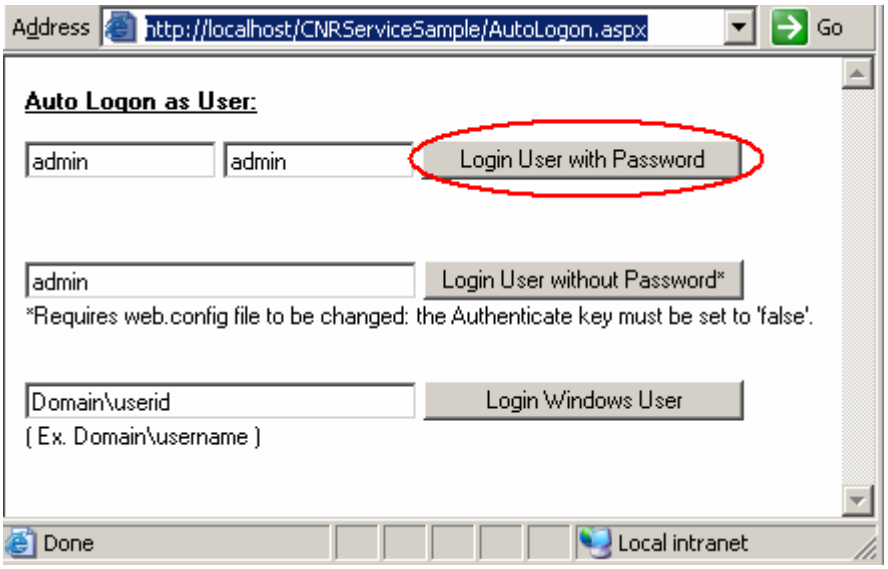
```
CNRService objService = new CNRService();
AuthHeader authentication = new AuthHeader();

authentication.Token = objService.AuthenticateWindows(winUserid.Text);
objService.AuthHeaderValue = authentication;
```

Now calls to the web service can be performed. These calls will write an entry to the Cizer .Net database with a unique ID. The command will be completed by passing this ID to the CNRServiceURLManager.aspx page in the Cizer.Net reporting application. An example of this will be seen in the section below.

Logging on to Cizer.Net

The web service allows users to be authenticated by a username and password combination, or by username only. In order to use the *AuthenticateNone*() function, the Authenticate key in the web.config must be set to *false*. The example below shows the three methods of authentication;

AuthenticateForms(), **AuthenticateNone()**, and **AuthenticateWindows()**.



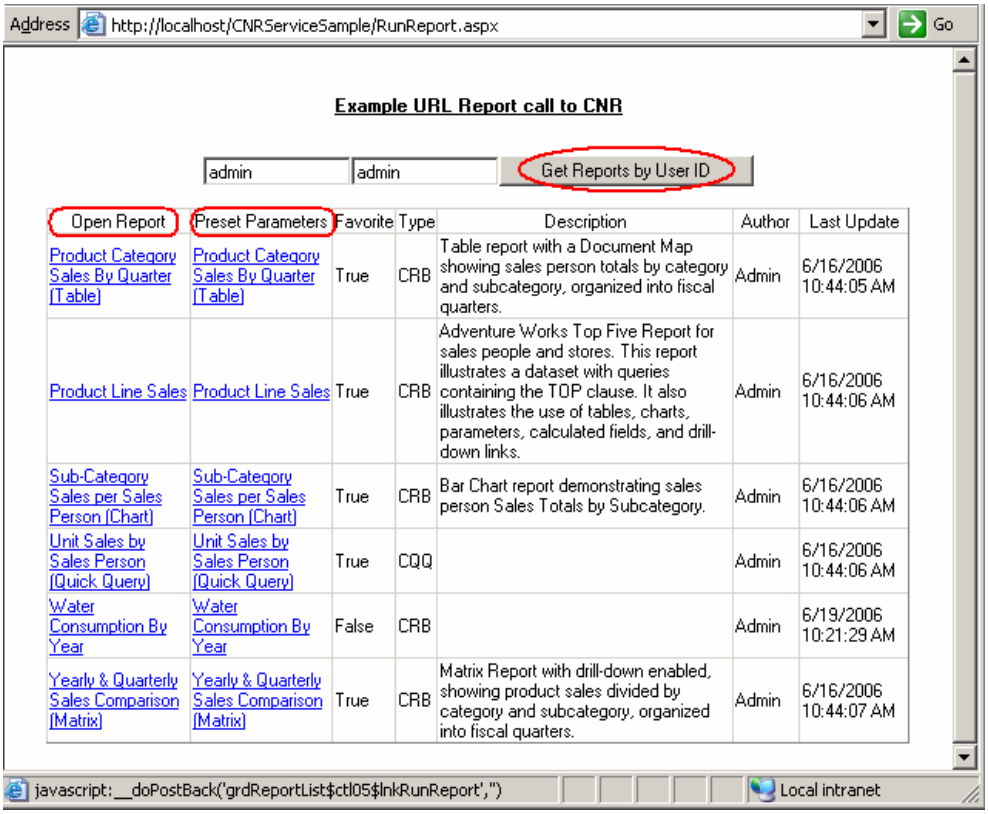
After calling one of these methods, you may decide to send the user to the main portal screen by calling the **DisplayPortal()** function, or to the Quick Query screen with the **DisplayQuickQuery()** command.

Here is an example of the entire process:

```
// STEP 1: Initialize the service and security
objService = new CNRService();
authentication = new AuthHeader();

// STEP 2: Login with the userid only
authentication.Token = objService.AuthenticateNone(userid.Text);
objService.AuthHeaderValue = authentication;

// STEP 3: Call the web service function to create the unique ID
objService.DisplayPortal();

// STEP 4: Send the ID to the URL Manager (to display the portal page)
Response.Redirect("http://localhost/cnr/cnrserviceurlmanager.aspx?
CNRServiceID=" + authentication.Token);
```

Retrieving Reports

In the Cizer .Net reporting application, certain users have access to certain reports. In order to retrieve a list of reports that a user has access to, the function **GetReportCollection()** is used. In the example above, this call would take place in "STEP 3." This is seen in the Sample application on the **RunReports.aspx** page (shown below).



Running Reports

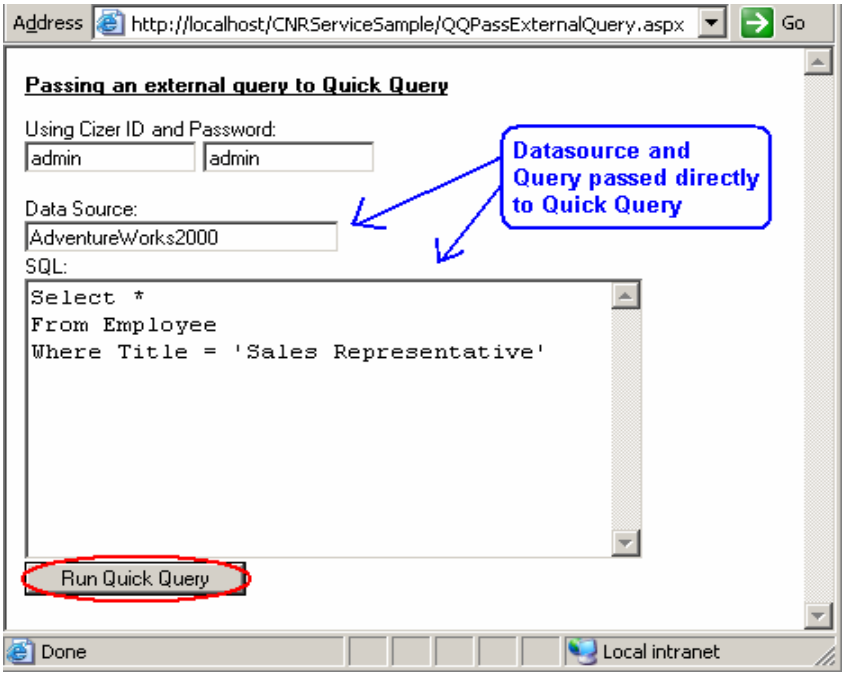
After retrieving the list of reports, the **RunReport()** command is called the run the report of choice. Once the authentication is set up, only the ID of the report is needed to run it. The code is as follows:

```
objService.RunReport(ReportID);
Response.Redirect("http://localhost/cnr/cnrserviceurlmanager.aspx?
CNRServiceID=" + authentication.Token);
```

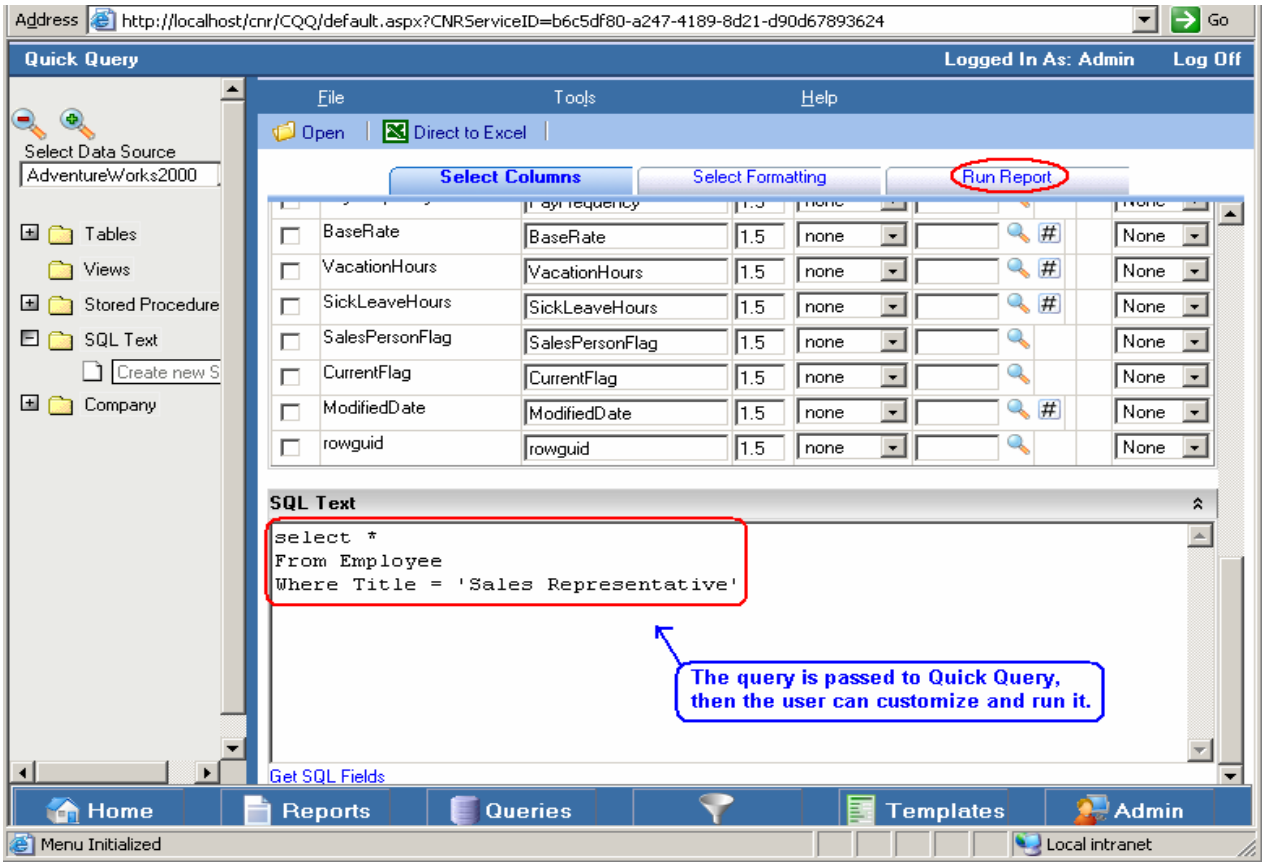
Passing Dynamic Elements

This new web service has the ability to make reports and queries dynamic by allowing dynamic elements, such as queries and parameters, to be passed directly into the Query page or into the Cizer.Net report page. This is seen in the Sample application in the **QQPassExternalQuery.aspx** and **RunReportWithPresetParameters.aspx** pages (shown below).

Dynamic Query: As seen below, given a user and a data source, any valid SQL statement can be passed directly to Quick Query for further visualization and customization.

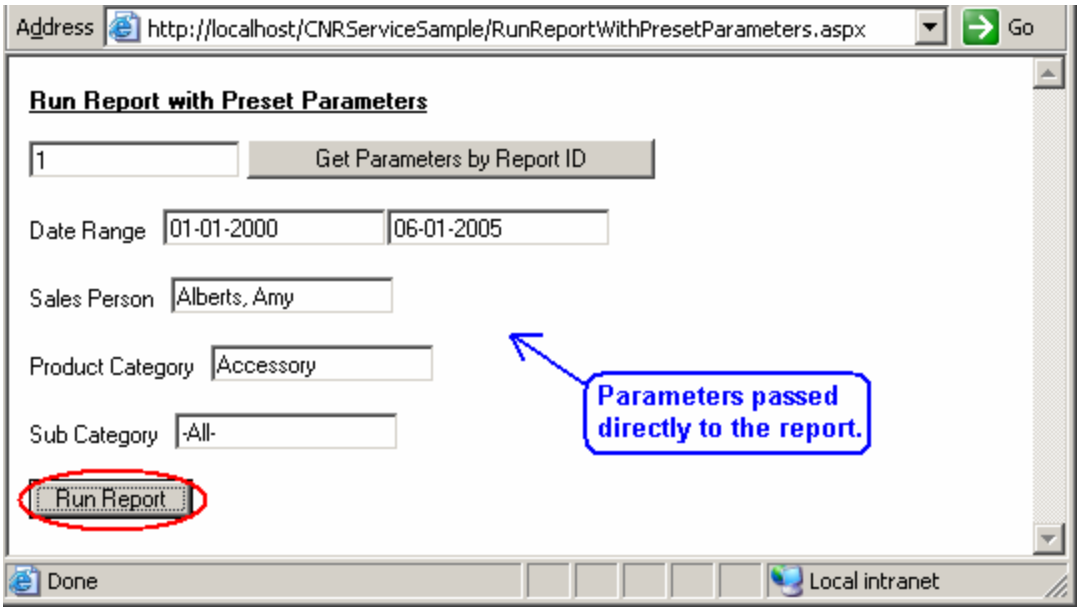


The user is directed to Quick Query with the query seen in the display (shown below).

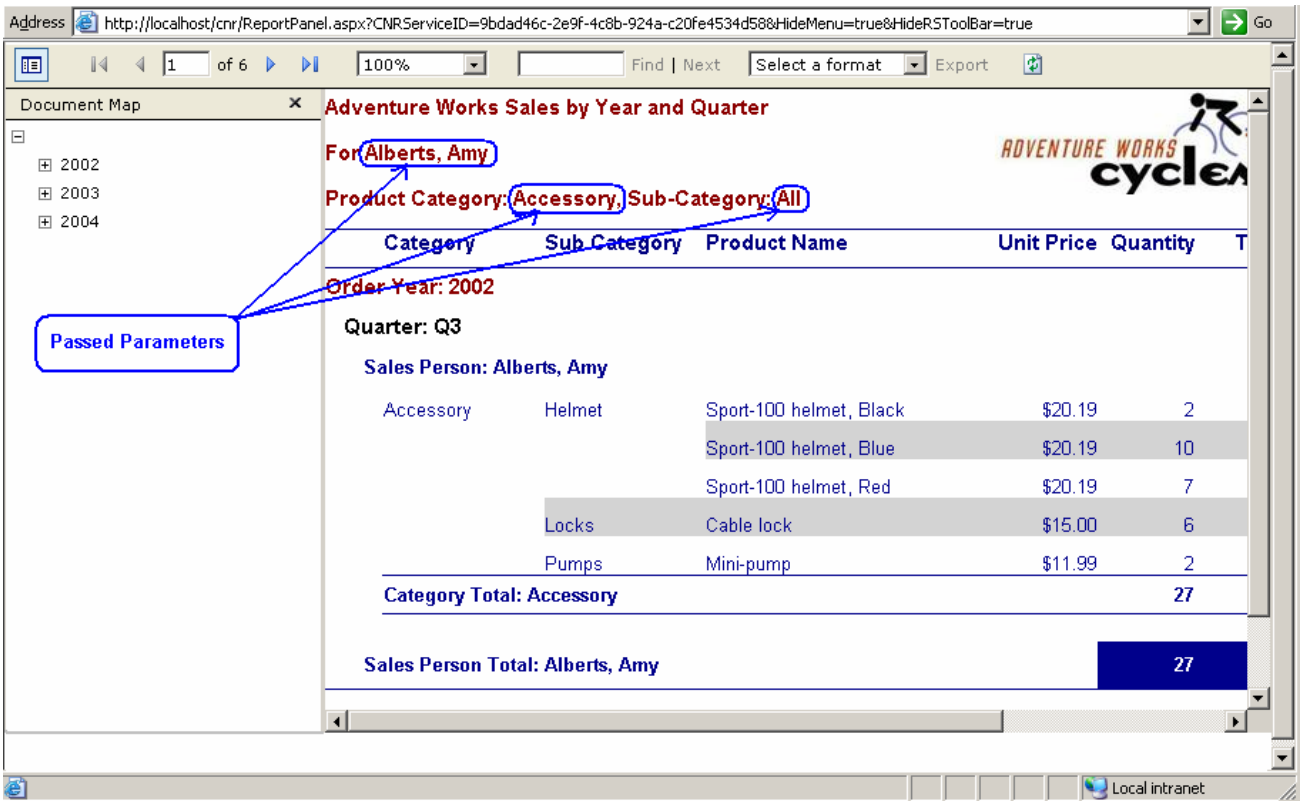


Dynamic Parameter: With the use of this service, dynamic values can be passed into Cizer.Net reports through pre-defined parameters.

The Sample application will allow you to test this functionality (shown below).



As seen below, the user is directed to the report, which has already been run using the set parameters.

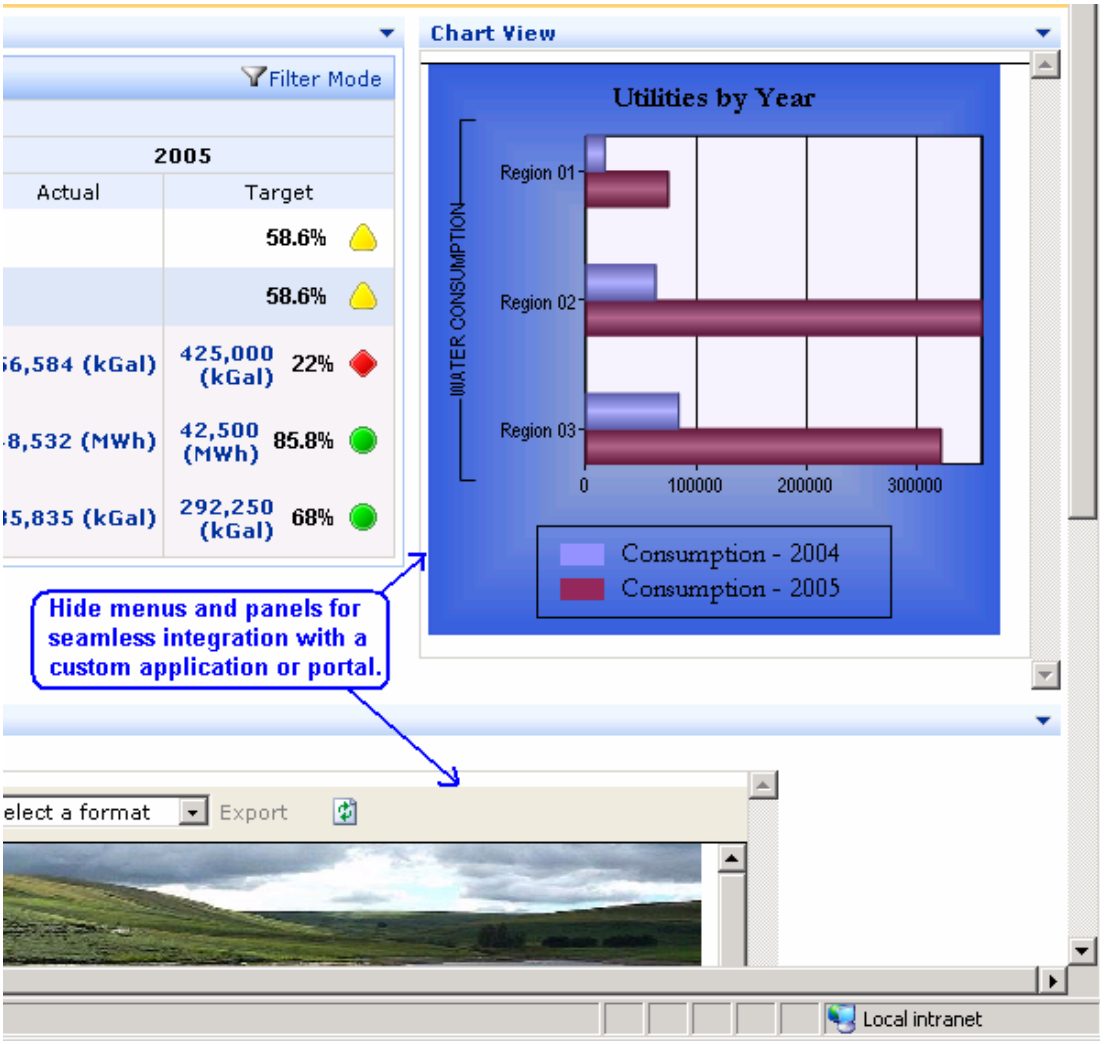


Display Options

Depending on the environment in which the Cizer.Net Web Service is used, certain parts of the portal page can be hidden. For example, a Cizer.Net report may just be part of a larger hosting application portal and may need to fit seamlessly within an IFrame of the hosting application. So, when redirecting the user in "STEP 4," parameters can be added to the end of the URL that will hide the Reporting Services Toolbar, as well as, the menus. For example, the call to the URL may look as follows:

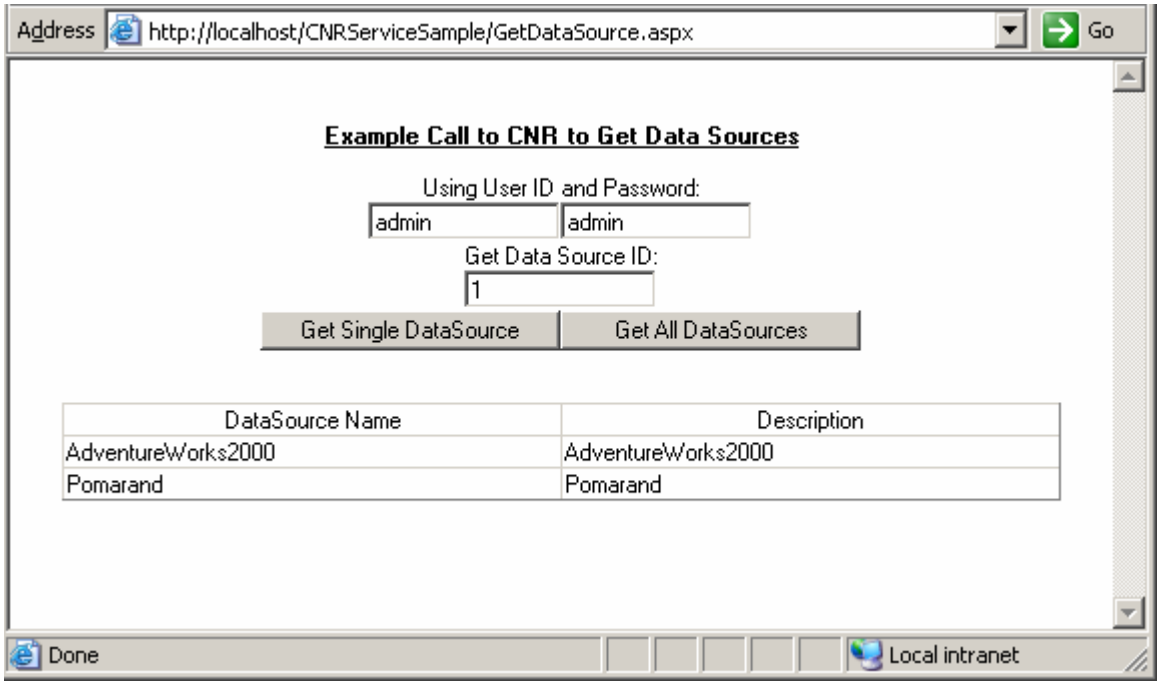
```
Response.Redirect("http://localhost/cnr/cnrserviceurlmanager.aspx?
HideMenu=true&HideRSToolBar=true&CNRServiceID="+ authentication.Token;
```

This will make it easier to incorporate Cizer .Net reports and charts into an existing custom application or portal page, as seen in the example below.



Retrieving Data Sources

Just as certain users have access to certain reports, access to Data Sources is restricted as well. In order to retrieve a list of Data Sources that a user has access to, the function **GetDataSourceCollection()** is used. The **GetDataSource()** function may also be used to get a single Data Source as specified by its unique ID. A clear example is seen in the Sample application on the **GetDataSource.aspx** page (shown below).



Conclusion

In customizing your solutions, Cizer .Net reporting can now be seamlessly embedded in any of your applications. The Sample application will help you understand the Cizer .Net Web Service to provide a fast and effective reporting solution to your customer.