

# Worldwide Customer Services

## Embedding Applications with Java

August 2019



Active Technologies, EDA, EDA/SQL, FIDEL, FOCUS, Information Builders, the Information Builders logo, iWay, iWay Software, Parlay, PC/FOCUS, RStat, Table Talk, Web390, WebFOCUS, WebFOCUS Active Technologies, and WebFOCUS Magnify are registered trademarks, and DataMigrator and Hyperstage are trademarks of Information Builders, Inc.

Adobe, the Adobe logo, Acrobat, Adobe Reader, Flash, Adobe Flash Builder, Flex, and PostScript are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Due to the nature of this material, this document refers to numerous hardware and software products by their trademarks. In most, if not all cases, these designations are claimed as trademarks or registered trademarks by their respective companies. It is not this publisher's intent to use any of these names generically. The reader is therefore cautioned to investigate all claimed trademark rights before using any of these names other than to refer to the product described.

Copyright © 2019, by Information Builders, Inc. and iWay Software. All rights reserved. Patent Pending. This manual, or parts thereof, may not be reproduced in any form without the written permission of Information Builders, Inc.

### Table of Contents

CONTENTS.....	3
EXECUTIVE SUMMARY .....	4
Roles .....	5
Setting Up Eclipse IDE .....	6
Importing JAR Files .....	6
Setting Up the Eclipse Tomcat Server .....	8
Running Your Web Application.....	11
Signing On .....	13
Regular Report .....	16
Deferred Report .....	25
ReportCaster .....	36
Signing Out.....	51
Possible Errors.....	52
Additional Resources .....	55

## EXECUTIVE SUMMARY

The goal of this project was to create an embedded web application sample that utilizes WebFOCUS RESTful web services using Java or Python. The embedded web application allows the user to sign in, run a report, view a deferred report, access schedules in ReportCaster, and sign out. The development period of this web application was five weeks: three weeks for programming, and two weeks for user experience development and testing.

Two versions of this web application were developed by separate teams using Java and Python, respectively. This documentation concerns the Java version of the web application sample. It was developed using the Eclipse IDE for Enterprise Java Developers (Java EE) in JavaServer Pages (JSP). For instructions on setting up the Eclipse environment and importing necessary Jar files in order to properly run the sample web application, see [Setting Up Eclipse IDE](#). Thank you for your interest in our sample application and we hope this documentation proves useful to you!

## Roles

The following people are the contributors and support resources who helped us deliver this project.

### Worldwide Customer Services

Name	Title
Joanne Han	Intern
Colin Salvato	Intern

### Other Information Builders Resources

Name	Department	Title
Alan Boord	Customer Support Services	Senior Support Manager
Jessica Chen	WebFOCUS Technical Division	Software Developer
Frances Gambino	Technical Content Management	Vice President, Customer Success
David Glick	WebFOCUS Product Management	Director
Ira Kaplan	WebFOCUS Product Management	Product Manager
Stefan Kostial	Technical Content Management	Director, Self-Service Learning
Tony Li	Customer Support Services	Senior Director, CSS and Premium Support
Breda McCrohan	Customer Support Services	Executive Administrator
Emily Nesson	Customer Support Services	Intern
Hamza Qureshi	Customer Support Services	Intern
Cheryl Simonsen	Corporate Admin. & Human Resources	HR & Talent Acquisition Specialist
Alain Theriault	Customer Support Services	Technical Support Rep III

## Setting Up Eclipse IDE

The Eclipse IDE for Enterprise Java Developers (also known as Eclipse IDE for Java EE) is similar to the basic Eclipse IDE version, except that it contains useful plugins for working with database and web development. For our Java web application sample, we use the Java EE version to implement and execute our code. If you do not have this version downloaded and would like to do so, navigate to the following link:

<https://www.eclipse.org/downloads/packages/release/2019-06/r/eclipse-ide-enterprise-java-developers>

The code itself is written in JavaServer Pages (JSP) files, which can contain both HTML and Java code. The Java code is able to generate dynamic content and can work with the regular static HTML content. This useful feature enables the resulting webpage to be dynamic.

### Importing JAR Files

This embedded web application sample requires Java Archive (JAR) files in order to successfully run. A JAR file is a zip file that can contain both compiled Java code (.class files) and Java source code (.java files). These files allow Java run times to efficiently deploy an entire application in a single request.

The JAR Files needed to successfully run the embedded web application are as follows:

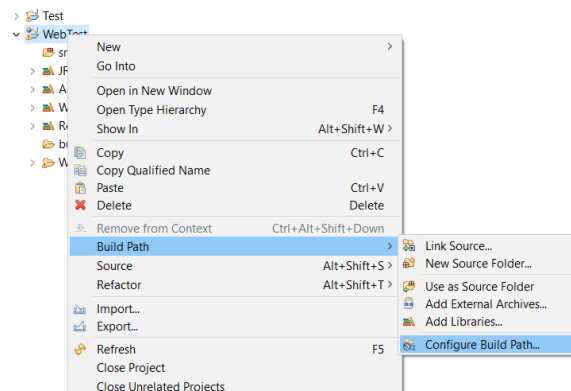
- commons-codec-1.9.jar
- commons-io-2.6-javadoc.jar
- commons-io-2.6-sources.jar
- commons-io-2.6-test-sources.jar
- commons-io-2.6-tests.jar
- commons-io-2.6.jar
- commons-logging-1.2.jar
- httpclient-4.4.jar
- httpcore-4.4.jar

Please refer to the following links to download the corresponding JAR files:

- To download the commons-io-2.6 jar files, navigate to the following link:  
[https://commons.apache.org/proper/commons-io/download\\_io.cgi](https://commons.apache.org/proper/commons-io/download_io.cgi)
- To download the commons-codec-1.9.jar file, navigate to the following link:  
<https://repo1.maven.org/maven2/commons-codec/commons-codec/1.9/>  
(scroll down to the commons-codec-1.9.jar file and click to download)
- To download the httpclient-4.4 jar file, navigate to the following link:  
<https://repo1.maven.org/maven2/org/apache/httpcomponents/httpclient/4.4/>  
(scroll down to the httpclient-4.4.jar file and click to download)
- To download the httpcore-4.4.jar file, navigate to the following link:  
<https://repo1.maven.org/maven2/org/apache/httpcomponents/httpcore/4.4/>  
(scroll down to the httpcore-4.4.jar file and click to download)

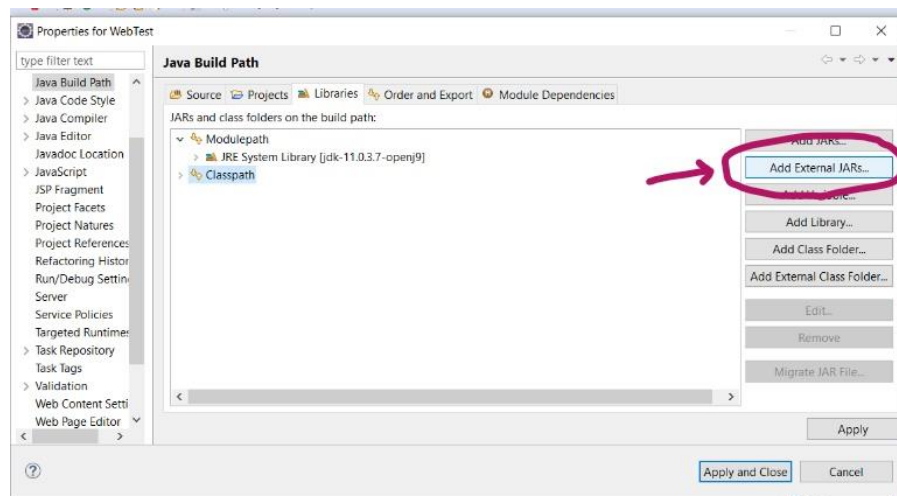
To import the JAR files into your Eclipse project, please refer to the following steps:

1. Right click the project folder, point to **Build Path**, and select **Configure Build Path**.

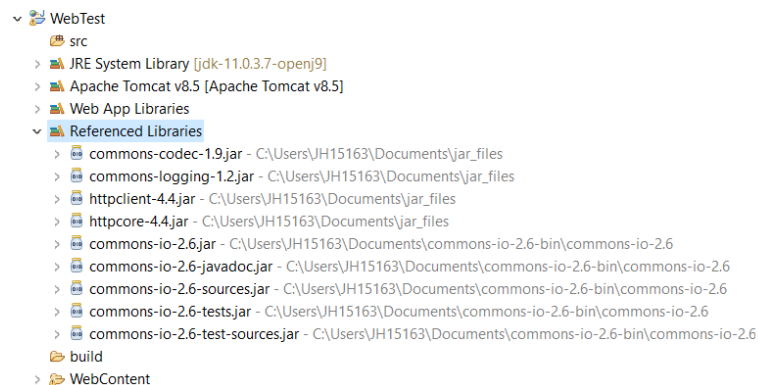


The Java Build Path dialog box displays.

2. In the Java Build Path dialog box, click **Classpath** and then click the **Add External JARs** button.



3. Locate the folder where your JAR files are stored, and open them in the Classpath. Once finished, click the **Apply and Close** button. In order to make sure your JAR files have been successfully imported, refer back to your project file in the Package Explorer view in Eclipse. To do so, click **Referenced Libraries** to expand the drop-down menu.



4. All of your imported JAR files should be listed. If they are not, you can manually select and drag your JAR files into the Referenced Libraries folder.

Congratulations! You have successfully imported the JAR files.

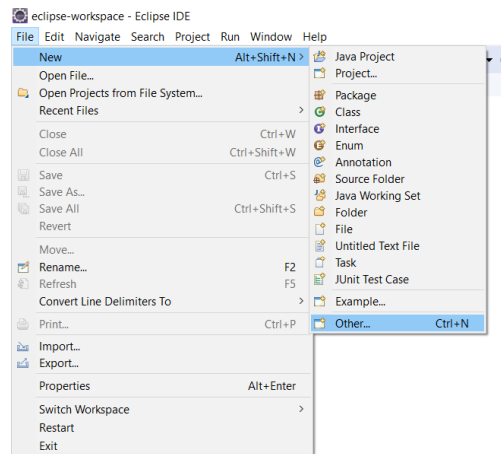
## Setting Up the Eclipse Tomcat Server

The Java EE version of Eclipse is able to run web applications through Apache Tomcat, an open source web server developed by the Apache Software Foundation that provides a web server environment in which Java code can run. The web application is able to run not only in Eclipse's own browser, but also in any third-party browser (for example, Chrome™, Firefox®, Microsoft® Edge®, and so on), as long as the Tomcat server in Eclipse is running. This is especially useful if you wish to debug the code or add new features on top of the sample web application.

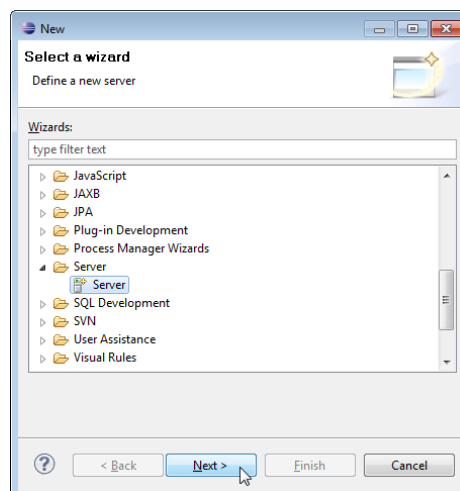
**Note:** You should already have Apache Tomcat installed in your computer beforehand, since you will be connecting Eclipse to that folder as you set up the server.

To set up the Tomcat Server in Eclipse, please refer to the following steps:

1. From the Eclipse main menu, click **File**, point to **New**, and select **Other**, as shown in the following image.

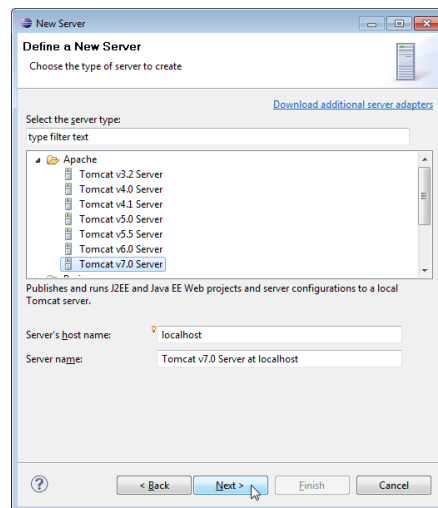


2. In the New dialog box, click the **Server** folder, select **Server**, and then click **Next**, as shown in the following image.





3. Select the Tomcat server version you would like to use and click **Next**, as shown in the following image.



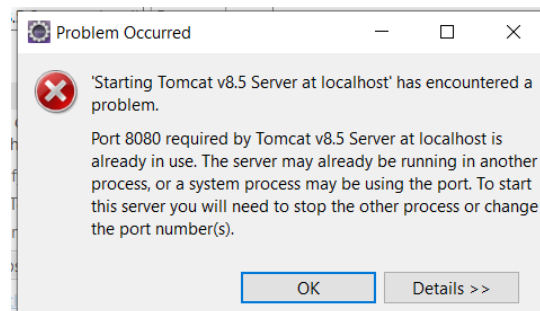
4. Navigate to the folder where you have Tomcat installed and select **Finish**.

Congratulations! You've successfully set up your Tomcat server in Eclipse.

By default, the Tomcat server is configured to the following port numbers:

- Tomcat admin port: 8005
- HTTP/1.1: 8080
- AJP/1.3: 8009

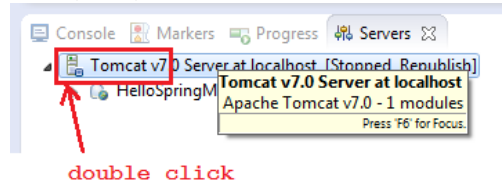
If your port numbers are the same as the ones used by WebFOCUS, you may run into the following error message:



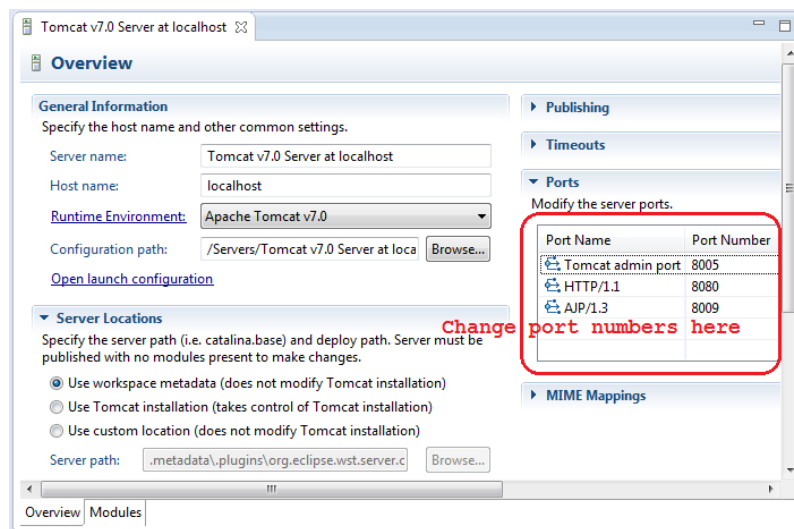
A formal definition of a port number is, “The logical address of each application or process that helps in identifying the sender and receiver processes of messages.” As an example, consider the following scenario: You have baked a cake for a friend, and you ask your sibling to deliver the cake to your friend’s house. In this case, you and your friend are two different computers, and your sibling is the connection between the two computers, using a network. Your sibling arrives at your friend’s house, but there are two different doors and they is not sure which one belongs to your friend’s. This is where port numbers would come in. The port number specifies which “door” to enter to deliver the message. In this case, the port numbers 8005, 8080, and 8009 are already used by WebFOCUS, so our web application cannot occupy those numbers.

In this scenario, you would have to change the port numbers for Eclipse, since the default ports are already used by WebFOCUS. In order to change the port numbers of the Tomcat server in Eclipse, refer to the following steps:

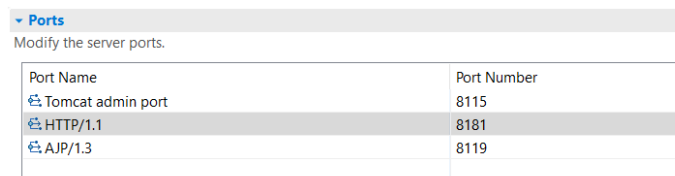
1. In the Servers view (usually located at the bottom where the **Console** view is), double click the server name, as shown in the following image.



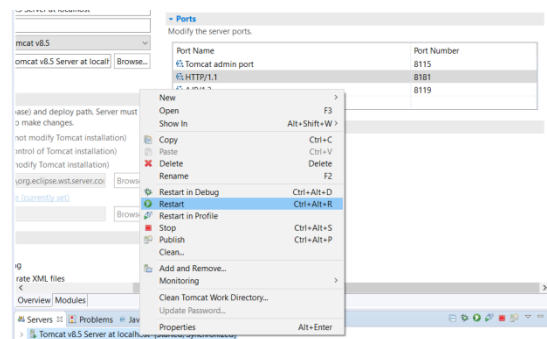
2. Once the configuration page for Tomcat is opened, notice the port numbers shown on the right, as shown in the following image.



3. Click the individual port numbers to change them to any number you would like. In the following example, the zeros have been replaced with ones for each port number:



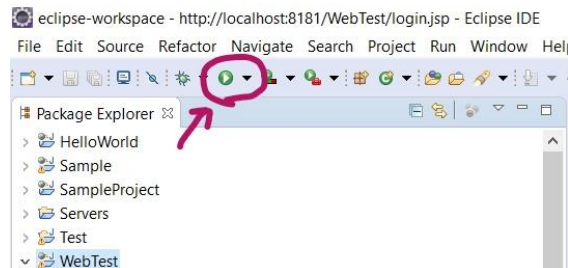
4. Press Ctrl + S to save your changes and right click the server in the **Server** view > click **Restart** to restart the server



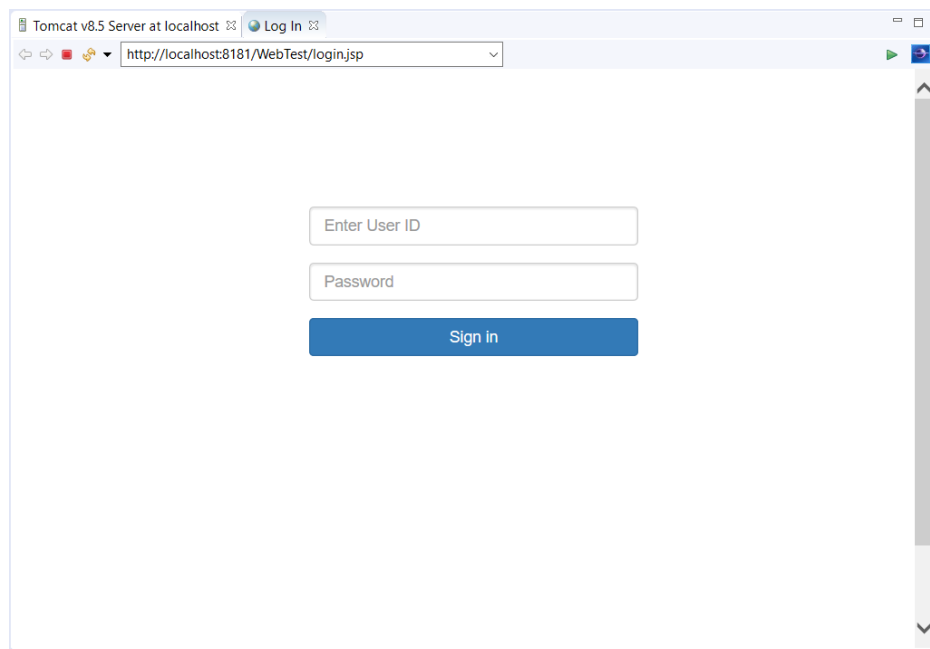
## Running Your Web Application

To run your web application via the Eclipse browser, please refer to the following steps:

1. Click the project folder of your web application, and from the ribbon, select the **Run** button.



2. Your web application should appear inside Eclipse's browser, as shown in the following image.



To deploy your web application using an third-party browser, such as Chrome, Firefox, and so on, refer to the following steps:

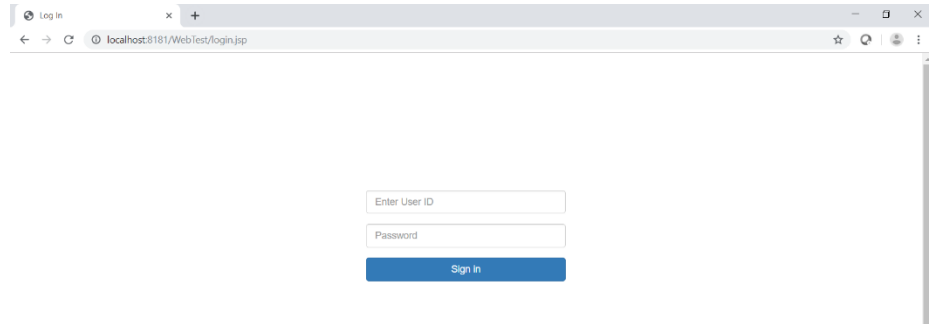
1. Type the following url into the browser of your choice:  
<http://localhost:portnumber/ProjectFolderName/login.jsp>
  - a. For the **portnumber**, use the number that is used for the HTTP/1.1 port:

▼ Ports

Modify the server ports.

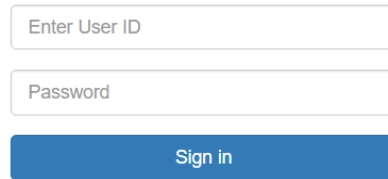
Port Name	Port Number
Tomcat admin port	8115
HTTP/1.1	8181
AJP/1.3	8119

- b. For the **ProjectFolderName**, use the name of the project folder of your web application. In this case, our **portnumber** (HTTP/1.1 number) is 8181, and the **ProjectFolderName** is WebTest, so the url we use is <http://localhost:8181/WebTest/login.jsp>.
2. Once you type the URL into the browser, press Enter, and the web application should run.



Congratulations! You have successfully run the web application.

## Signing On



Enter User ID

Password

Sign in

The user can sign in to the web application using the **admin** for the user name and **admin** for the password.

The sign in is executed using the corresponding REST call, as well as the login.jsp and logincheck.jsp files.

### REST Call and URL Format

The REST call made and its details to execute the sign in are as follows:

**HTTP Method:** POST

**REST URL Format:** [http://host:port/ibi\\_apps/rs/ibfs](http://host:port/ibi_apps/rs/ibfs)

where:

**host**

Is the name of the system where WebFOCUS is installed.

**port**

Is the port number used by WebFOCUS.

**Body Format:**

**IBIRS\_action=SignOn**

### Code & JSP Files

LOGIN.JSP: This file creates the webpage where users enter their credentials in order to sign in.

```
<!DOCTYPE html>

<html lang="en">

<head>
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>Log In</title>
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
  <link rel="stylesheet" href="css/log-in.css">
</head>

<body>

  <div class="container" id="mainContainer">
```

```

        <form class="form" role="form" method="post" action="Logincheck.jsp" accept-
charset="UTF-8" id="Login-nav">
            <div class="form-group">
                <label class="sr-only" for="enterUserId">User ID</label>
                <input type="text" class="form-control" id="enterUserId"
name="enterUserId" placeholder="Enter User ID" required>
            </div>
            <div class="form-group">
                <label class="sr-only" for="userPassword">Password</label>
                <input type="password" class="form-control" id="userPassword"
name="userPassword" placeholder="Password" required>
            </div>
            <div class="form-group">
                <button type="submit" class="btn btn-primary btn-block">Sign
in</button>
            </div>
        </form>

    </div>

</body>

</html>

```

LOGINCHECK.JSP: This file checks the users credentials entered into login.jsp.

```

<%@ page
import="java.util.*,
    java.awt.Frame,
    java.io.BufferedReader,
    java.io.File,
    java.io.FileOutputStream,
    java.io.IOException,
    java.io.InputStream,
    java.io.InputStreamReader,
    java.io.PrintWriter,
    java.io.UnsupportedEncodingException,
    java.util.ArrayList,
    org.apache.http.Header,
    org.apache.http.HttpResponse,
    org.apache.http.NameValuePair,
    org.apache.http.client.*,
    org.apache.http.client.entity.UrlEncodedFormEntity,
    org.apache.http.client.methods.*,
    org.apache.http.impl.client.CloseableHttpClient,
    org.apache.http.impl.client.HttpClients,
    org.apache.http.message.BasicNameValuePair,
    java.net.InetAddress,
    java.net.UnknownHostException"%>

<%
/**
 * Variable hostname is defined and then pulled from the user's computer
 * Variable addr is defined and then pulled from the user's computer
 * Exceptions are caught if the program can't access the users hostname
 */
String hostname = "Unknown";

try {
InetAddress addr;

```

```

addr = InetAddress.getLocalHost();
hostname = addr.getHostName();
} catch (UnknownHostException ex) {
System.out.println("Hostname cannot be resolved");
}

/**
 * This is done in order to maintain a session
 * The cookies array stores the cookies so they can be maintained on other webpages
 * A client is also opened in order to maintain the session
 */
Header[] cookies = null;
CloseableHttpClient client = HttpClients.createDefault();

/**
 * String url is constructed using the hostname
 */
String url = "http://" + hostname + ".ibi.com:8080/ibi_apps/rs/ibfs";
HttpPost method = new HttpPost(url);

/**
 * NameValuePair[] is made in order to save the users credentials and maintain the
session
 */
ArrayList<NameValuePair> parameters = new ArrayList<NameValuePair>();
parameters.add(new BasicNameValuePair("IBIRS_action", "signOn"));
parameters.add(new BasicNameValuePair("IBIRS_userName", "admin"));
parameters.add(new BasicNameValuePair("IBIRS_password", "admin"));

/**
 * This try catch is there to catch an error for a missing parameter saved in the above
arraylist
 */
try {
method.setEntity(new UrlEncodedFormEntity(parameters));
} catch (Exception ex) {
System.out.println("Set Parameters Error: "+ex.getMessage());
}

/**
 * Sets httpResponse equal to the post method
 * Saves the cookies for use on other pages
 * Sets session attributes for use on other pages
 */
try {
HttpResponse httpResponse = client.execute(method);
int statusCode = httpResponse.getStatusLine().getStatusCode();

cookies = method.getHeaders("Set-Cookie");
cookies = httpResponse.getHeaders("Set-Cookie");

session.setAttribute("cookies", cookies);
session.setAttribute("client", client);
session.setAttribute("user", "IBIRS_userName");
session.setAttribute("hostname", hostname);
response.sendRedirect("index.jsp");
} catch (Exception ex) {
System.out.println("HttpResponse: "+ex.getMessage());
}
%>

```

## Regular Report

[Company ABC](#) [Home](#) [Regular Report](#) [Deferred Report](#) [ReportCaster](#) [Log Out](#)

### Available Reports

Click to view your reports!

Report Name	Action
Delayed Shipments Report	<a href="#">View Report</a>
Product Revenue Report	<a href="#">View Report</a>
Delayed Shipments Chart	<a href="#">View Report</a>
Gross Profit Chart	<a href="#">View Report</a>

Shipping Company	Days Delayed
DL Express	4,490
Express Star	137,813
Logistics Star	18,830
Parcel Service United	187,700
Postal Service	62,505

The user can choose from a number of reports in the Public folder in the WebFOCUS Repository. The names of the reports are listed in the table, and appear in the iFrame.

The report request is executed using the corresponding REST call, as well as the regular-report.jsp and regular-redirect.jsp files:

### REST Call and URL Format

The REST call made to GET the Public Folder in the WebFOCUS Repository and its contents is as follows:

**HTTP Method:** GET

**REST URL Format:** [http://host:port/ibi\\_apps/rs/ibfs/wfc/Repository/FolderName](http://host:port/ibi_apps/rs/ibfs/wfc/Repository/FolderName)

where:

**host**

Is the name of the system where WebFOCUS is installed.

**port**

Is the port number used by WebFOCUS.

**FolderName**

Is the name of the folder used for the stored WebFOCUS report. If the folder used for the stored WebFOCUS report exists as a subfolder, then the path to the subfolder name must be included in the REST URL. For example, TopFolderName/SubFolderName.

**Body Format:**

**IBIRS\_action=get**

The REST call made to GET the specified report and run it is as follows:

**HTTP Method:** GET

**REST URL Format:** [http://host:port/ibi\\_apps/rs/ibfs/wfc/Repository/FolderName/reportName](http://host:port/ibi_apps/rs/ibfs/wfc/Repository/FolderName/reportName)

where:

**host**

Is the name of the system where WebFOCUS is installed.

**port**

Is the port number used by WebFOCUS.

**FolderName**



Is the name of the folder used for the stored WebFOCUS report. If the folder used for the stored WebFOCUS report exists as a subfolder, then the path to the subfolder name must be included in the REST URL. For example, TopFolderName/SubFolderName.

reportName

Is the name of the selected WebFOCUS report.

#### Body Format:

IBIRS\_action=run&IBIRS\_service=ibfs

## Code & JSP Files

REGULAR-REPORT.JSP: This file displays the webpage that has a table displaying titles of all accessible reports and buttons allowing the user to view said reports in the iFrame.

```
<!DOCTYPE html>

<%@ page
    import="java.util.*,
            java.net.*,
            java.io.*,
            java.awt.Frame,
            java.io.BufferedReader,
            java.io.File,
            java.io.FileOutputStream,
            java.io.IOException,
            java.io.InputStream,
            java.io.InputStreamReader,
            java.io.PrintWriter,
            java.io.UnsupportedEncodingException,
            java.util.ArrayList,
            org.apache.http.Header,
            org.apache.http.HttpResponse,
            org.apache.http.NameValuePair,
            org.apache.http.client.*,
            org.apache.http.client.entity.UrlEncodedFormEntity,
            org.apache.http.client.methods.*,
            org.apache.http.impl.client.CloseableHttpClient,
            org.apache.http.impl.client.HttpClients,
            org.apache.http.message.BasicNameValuePair"%>

<%
    /**
     * Redirects the user to the login page if they no longer have an active session
     */
    String user = (String) session.getAttribute("user");
    if (user == null) {
        response.sendRedirect("login.jsp");
    } else {
%>

<%
    /**
     * Defines hostname, cookies[], client, which are pulled from logincheck.jsp to maintain
    the session
     * Defines arrayLists to hold urls, report titles, and report names
     */
```

```

String hostname = (String) session.getAttribute("hostname");
Header[] cookies = (Header[]) session.getAttribute("cookies");
CloseableHttpClient client = (CloseableHttpClient) session.getAttribute("client");
List<String> urls = new ArrayList<String>();
List<String> reportTitles = new ArrayList<String>();
List<String> reportNames = new ArrayList<String>();

/**
 * folderURL is put together using hostname to access the XML file for the public
folder
 */
String folderURL = "http://" + hostname
+
".ibi.com:8080/ibi_apps/rs/ibfs/WFC/Repository/Public?IBIRS_action=get";

/**
 * Defines getFolderContent as get request using the folderURL
 */
HttpGet getFolderContent = new HttpGet(folderURL);

/**
 * For loop attaches cookies to "getFolderContent"
 */
for (int h = 0; h < cookies.length; h++) {
    getFolderContent.addHeader(cookies[h].getName(), cookies[h].getValue());
}

/**
 * Defines HttpResponse outside of the try block so it can be called outside of it
 */
HttpResponse httpResponse = null;

/**
 * Executes getFolderContent
 * Gets the status code for possible errors
 * Catch displays possible error messages
 */
try {
    httpResponse = client.execute(getFolderContent);
    int statusCode = httpResponse.getStatusLine().getStatusCode();
} catch (Exception ex) {
    System.out.println("HttpResponse Error: " + ex.getMessage());
}

/**
 * Defines HttpResponse outside of the try block so it can be called outside of it
 */
try {
    InputStream rstream = httpResponse.getEntity().getContent();
    BufferedReader br = new BufferedReader(new InputStreamReader(rstream));
    String line;
    String newOutput = null;

    /**
     * While loop reads the webpage line by line
     * Each line is concatenated into "line"
     */
    while ((line = br.readLine()) != null) {
        newOutput = line;
    }
    /**
     * "line" is split up by "<" into an array "stringArray"

```

```

        * Each line is checked to see if it contains "description=" &
typeDescription="report"
        * If it fits that criteria it is saved to "descLine"
        */
String[] stringArray = line.split("<");
for (int i = 0; i < stringArray.length; i++) {
    if (stringArray[i].contains("description=\"")
        &&
stringArray[i].contains("typeDescription=\"Report\"")) {
        String descLine = stringArray[i];
        /**
         * descLine is split up by spaces into stringArray2[]
         * for loop iterates through the strings to see if they
contain "description"

         * if they do, they are saved to reportTitles[]
         */
String[] stringArray2 = descLine.split("\\s+");
for (int j = 0; j < stringArray2.length; j++) {
    String descLine2 = stringArray2[j];
    if (descLine2.contains("description")) {
        String descLine3 =
descLine2.replace("description=\"", "").replace("\\", "");
        reportTitles.add(descLine3);
    }
    /**
     * if descLine2 line contains "name=" then the
name is saved to "reportNames[]"

     */
    if (descLine2.contains("name=\"")) {
        String nameLines =
descLine2.replace("name=\"", "").replace("\\", "");
        reportNames.add(nameLines);
    }
}
    }
}

/**
 * The buffered reader is closed
 * Catch makes sure to keep any errors from executing and displays the error
message

 */
br.close();
} catch (Exception ex) {
    System.out.println("RStream Error: " + ex.getMessage());
}

/**
 * urls array is iterated through and each string in it is split by "/"
 * From the url, the reportTitle is pulled and saved into reportTitles[]
 */
for (int i = 0; i < urls.size(); i++) {
    String[] sub = urls.get(i).split("/");
    for (int j = 0; j < sub.length; j++) {
        if (sub[j].contains(".fex")) {
            reportTitles.add(sub[j].replace(".fex", ""));
        }
    }
}
}

%>

```

```

<html lang="en">

<head>
<meta name="viewport" content="width=device-width, initial-scale=1">
<title>Regular Report</title>

<link rel="stylesheet"
      href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
<link rel="stylesheet" href="css/nav-css.css">
<link rel="stylesheet" href="css/sales-portal.css">

<script
      src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min.js"></script>
<script
      src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>

<script type="text/javascript">
    function clearFrame() {
        document.getElementById("theFrame").className = "noframe";
        document.getElementById("theFrameDiv").style.background = "url(https://mir-s3-cdn-
cf.behance.net/project_modules/disp/09b24e31234507.564a1d23c07b4.gif) center center no-
repeat";
    }
    function loadFrame() {
        document.getElementById("theFrame").className = "showframe";
        document.getElementById("theFrameDiv").style.background = "url(white_background.png)";
    }
</script>

<style>
.noframe {
    display: none;
}

.showframe {
    display: block;
}

.theFrame {
    height: 350px;
}

.button {
    background-color: #2496EF;
    border: none;
    color: white;
    padding: 8px 32px;
    text-align: center;
    text-decoration: none;
    display: inline-block;
    font-size: 16px;
    margin: 4px 2px;
    cursor: pointer;
    width: 80%;
    border-radius: 4px;
    transition: all 0.2s ease-in-out;
}

.button:hover {
    background-color: #79bffa;
}

```

```

}

th, tr {
    text-align: center;
}

td {
    width: 68%;
}

table {
    border: 1px solid black;
    overflow: scroll;
    border-collapse: collapse;
    width: 100%;
}

#tabdiv {
    display: block;
    height: 200px;
    overflow-y: scroll;
    overflow: auto;
}

tr:nth-child(even) {
    background-color: #f2f2f2;
}

form {
    width: 100%;
}

#header {
    text-align: center;
}
</style>
</head>

<body>
    <div class="container" id="mainContainer">
        <nav class="navbar navbar-default" id="page-nav">
            <div class="container-fluid">
                <div class="navbar-header">
                    <button type="button" class="navbar-toggle" data-
toggle="collapse"
                    data-target="#myNavbar">
                        <span class="icon-bar"></span> <span class="icon-
bar"></span> <span
                            class="icon-bar"></span>
                    </button>
                    <a class="navbar-brand" href="index.jsp">Company ABC</a>
                </div>
                <div class="collapse navbar-collapse" id="myNavbar">
                    <ul class="nav navbar-nav">
                        <li><a href="index.jsp">Home</a></li>
                        <li class="active"><a href="#">Regular Report</a></li>
                        <li><a href="deferred.jsp">Deferred Report</a></li>
                        <li><a href="reportcaster.jsp">ReportCaster</a></li>
                    </ul>
                    <ul class="nav navbar-nav navbar-right">
                        <li><a href="Logout.jsp" id="Log-out"><span

```

```

out"></span> Log Out</a></li>
                                </ul>
                            </div>
                        </div>
</nav>

<div id="header">
    <h2>Available Reports</h2>
    <p>Click to view your reports!</p>
    <br></div>

<table>
    <tr>
        <th class="titletext">Report Name</th>
        <th class="titletext">Action</th>
    </tr>
</table>

<div id=tabdiv>
    <table>
        <tr>
            <%
                for (int i = 0; i < reportTitles.size(); i++) {
            %>
            <td class="titletext"><%=reportTitles.get(i)%></td>
            <td>
                <form name="x" action="regular-redirect.jsp"
                    target="name_of_iframe" onclick="clearFrame()">
                    <input class="button" type="hidden" name="x"
                        value="<%=reportNames.get(i)%>"></input>
                    <button type="submit" class="button">View
Report</button>
                </form>
            </td>
        </tr>
        <%
            }
        %>
    </table>
</div>

<br> <br>

<div class="theFrame" id="theFrameDiv">
    <iframe style="border: none;" onLoad="loadFrame()"
        name="name_of_iframe" id="theFrame" width="100%"
height="350px"></iframe>
</div>

</div>
</body>

</html>

<%
}
%>

```

REGULAR-REDIRECT.JSP: This file executes the REST calls once the “View Report” button is clicked and feeds the HTML response back to the iFrame of the appropriate report.

```
<%@ page
import="java.util.*,
    java.net.*,
    java.io.*,
    java.awt.Frame,
    java.io.BufferedReader,
    java.io.File,
    java.io.FileOutputStream,
    java.io.IOException,
    java.io.InputStream,
    java.io.InputStreamReader,
    java.io.PrintWriter,
    java.io.UnsupportedEncodingException,
    java.nio.charset.StandardCharsets,
    java.util.ArrayList,
    org.apache.http.Header,
    org.apache.http.HttpResponse,
    org.apache.http.NameValuePair,
    org.apache.http.client.*,
    org.apache.http.client.entity.UrlEncodedFormEntity,
    org.apache.http.client.methods.*,
    org.apache.http.impl.client.CloseableHttpClient,
    org.apache.http.impl.client.HttpClients,
    org.apache.http.message.BasicNameValuePair,
    org.apache.commons.io.IOUtils"%>

<%
/**
 * Defines hostname, cookies[], client, which are pulled from logincheck.jsp to maintain
the session
 * Defines reportName by pulling it from the form block in regular-report.jsp
 */
String hostname = (String) session.getAttribute("hostname");
Header[] cookies = (Header[]) session.getAttribute("cookies");
CloseableHttpClient client = (CloseableHttpClient) session.getAttribute("client");
String reportName = request.getParameter("x");

/**
 * Defines reportURL to access the reports, using the hostname and reportName
 */
String reportURL = "http://" + hostname +
".ibi.com:8080/ibi_apps/rs?IBIRS_path=WFC/Repository/Public/"
    + reportName + "&IBIRS_action=run&IBIRS_service=ibfs";

/**
 * getReport is defined to send the get request using reportURL
 */
HttpGet getReport = new HttpGet(reportURL);

/**
 * For loop attaches the cookies to the get request
 */
for (int h = 0; h < cookies.length; h++) {
    getReport.addHeader(cookies[h].getName(), cookies[h].getValue());
}

/**
```

```

    * Defines HttpResponse and replaceString outside the try block
    */
    HttpResponse httpResponse = null;
    String replacedString = "";

    /**
     * httpResponse is set to execute getReport
     * Gets status code for possible error message
     */
    try {
        httpResponse = client.execute(getReport);
        int statusCode = httpResponse.getStatusLine().getStatusCode();
        String result = IOUtils.toString(httpResponse.getEntity().getContent(),
StandardCharsets.UTF_8);
        replacedString = result.replace("/ibi_apps/", "http://" + hostname +
".ibi.com:8080/ibi_apps/");
    } catch (Exception ex) {
        System.out.println("HttpResponse Error: " + ex.getMessage());
    }

    /**
     * Sends the report back to the iFrame
     */
%>

<%=replacedString%>

```



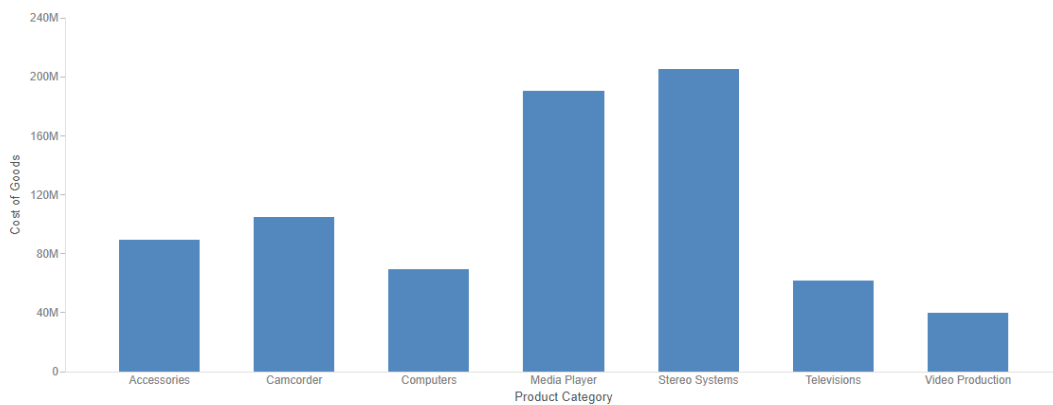
## Deferred Report

[Company ABC](#)[Home](#)[Regular Report](#)[Deferred Report](#)[ReportCaster](#)[Log Out](#)

### Deferred Reports

View the status of your deferred reports and click to view them!

Report Name	Report Status	Action
Delayed Shipments Chart	COMPLETE	<a href="#">View Report</a>
Delayed Shipments Report	COMPLETE	<a href="#">View Report</a>
Cost of Products Chart	COMPLETE	<a href="#">View Report</a>



The user can select from a number of deferred reports in the Public folder from the WebFOCUS Repository. These reports are listed in a table with their status and a View Report button for each. Clicking this button will display the appropriate report in the iFrame below the table.

The deferred report request is executed via its corresponding REST call as well as the deferred.jsp and defer-redirect.jsp files (as shown below):

### REST Call and URL Format

The REST call made to GET all the tickets of the available deferred reports is as follows:

**HTTP Method:** GET

**REST URL Format:** `http://host:port/ibi_apps/rs`

where:

`host`

Is the name of the system where WebFOCUS is installed.

`port`

Is the port number used by WebFOCUS.

**Body Format:**

`IBIRS_action=listTickets&IBIRS_service=defer`

The REST call made to GET and run the specified deferred report is as follows:

**HTTP Method:** GET

**REST URL Format:** `http://host:port/ibi_apps/rs`

where:

`host`

Is the name of the system where WebFOCUS is installed.

`port`

Is the port number used by WebFOCUS.

#### Body Format:

`IBIRS_action=getReport&IBIRS_ticketName=ticketName&IBIRS_service=defer`

where:

`ticketName`

Is the ticket of the specific deferred report

## Code & JSP Files

Deferred.jsp: This file displays the webpage that has a table displaying titles and status' of each of the deferred reports along with their corresponding "View Report" buttons.

```
<!DOCTYPE html>
```

```
<%@ page
```

```
import="java.util.*,
    java.net.*,
    java.io.*,
    java.awt.Frame,
    java.io.BufferedReader,
    java.io.File,
    java.io.FileOutputStream,
    java.io.IOException,
    java.io.InputStream,
    java.io.InputStreamReader,
    java.io.PrintWriter,
    java.io.UnsupportedEncodingException,
    java.util.ArrayList,
    org.apache.http.Header,
    org.apache.http.HttpResponse,
    org.apache.http.NameValuePair,
    org.apache.http.client.*,
    org.apache.http.client.entity.UrlEncodedFormEntity,
    org.apache.http.client.methods.*,
    org.apache.http.impl.client.CloseableHttpClient,
    org.apache.http.impl.client.HttpClients,
    org.apache.http.message.BasicNameValuePair"%>
```

```
<%
```

```
/**
 * Redirects the user to the login page if they don't have an active session
 */
String user = (String) session.getAttribute("user");
if (user == null) {
    response.sendRedirect("login.jsp");
} else {
```

```
%>
```

```
<%
```

```
/**
 * Defines hostname, cookies[], client, which are pulled from logincheck.jsp to maintain
the session
 */
String hostname = (String) session.getAttribute("hostname");
Header[] cookies = (Header[]) session.getAttribute("cookies");
```

```

CloseableHttpClient client = (CloseableHttpClient) session.getAttribute("client");

/**
 * Defines ArrayLists: reportNames, reportStatus, reportTickets
 */
List<String> reportNames = new ArrayList<String>();
List<String> reportStatus = new ArrayList<String>();
List<String> reportTickets = new ArrayList<String>();

/**
 * Creates the listTicketsURL using the user's hostname
 */
String listTicketsURL = "http://" + hostname
                        +
".ibi.com:8080/ibi_apps/rs?IBIRS_action=listTickets&IBIRS_service=defer";

/**
 * Defines getCall as get request using the listTicketsURL
 */
HttpGet getCall = new HttpGet(listTicketsURL);

/**
 * For loop attaches cookies to the "getCall" get request
 */
for (int h = 0; h < cookies.length; h++) {
    getCall.addHeader(cookies[h].getName(), cookies[h].getValue());
}

/**
 * Defines httpResponse outside the try block so it can be called outside the try
block
 */
HttpResponse httpResponse = null;

/**
 * Executes the get request
 * Gets status code for possible error
 * Catch prevents from executing any errors and displays error message
 */
try {
    httpResponse = client.execute(getCall);
    int statusCode = httpResponse.getStatusLine().getStatusCode();
} catch (Exception ex) {
    System.out.println("HttpResponse Error: " + ex.getMessage());
}

/**
 * Creates rstream and br to read the XML
 */
try {
    InputStream rstream = httpResponse.getEntity().getContent();
    BufferedReader br = new BufferedReader(new InputStreamReader(rstream));
    /**
     * Defines string line outside the while loop for use outside of it
     */
    String line;
    /**
     * While loop reads through the XML line by line
     */
    while ((line = br.readLine()) != null) {

```

```

String[] tempNameArray;
String tempName;
String[] tempNameArrayStatus;
String tempStatus;
/**
 * Splits up each line by "<"
 * For loop iterates through all lines
 * Checks if particular line contains: item _jt="IBFSMRDefTicketObject"
 * If so, that line is broken down by spaces
 * Iterates through those lines to get the report name and is added to
reportNames[]

 */
String[] stringArray = line.split("<");
for (int i = 0; i < stringArray.length; i++) {
    if (stringArray[i].contains("item
_jt=\"IBFSMRDefTicketObject\"")) {
        tempNameArray = stringArray[i].split("\\s+");
        for (int j = 0; j < tempNameArray.length; j++) {
            if (tempNameArray[j].contains("description=")) {
                tempName =
tempNameArray[j].replace("description=\"", "");
                reportNames.add(tempName);
            }
        }
    }
}
/**
 * Checks for lines if they contain "CTH_DEFER_"
 * Those lines are saved and then split by spaces
 * Pulls out the deferred report status
 * If the status equals "CTH_DEFER_READY" then it is saved to
the reportStatus array as "COMPLETE" otherwise "NOT COMPLETE"
 * Buffered reader is closed
 * Catch displays the error status message
 */
if (stringArray[i].contains("CTH_DEFER_")) {
    tempNameArrayStatus = stringArray[i].split("\\s+");
    for (int j = 0; j < tempNameArrayStatus.length; j++) {
        if
(tempNameArrayStatus[j].contains("name=\"CTH_DEFER_")) {
            tempStatus =
tempNameArrayStatus[j].replace("name=\"", "");
            tempStatus = tempStatus.replace("\"/>",
            "");
            if (tempStatus.equals("CTH_DEFER_READY"))
            {
                tempStatus = "COMPLETE";
            } else {
                tempStatus = "NOT COMPLETE";
            }
            reportStatus.add(tempStatus);
        }
    }
}

}
br.close();
} catch (Exception ex) {
    System.out.println("RStream Error: " + ex.getMessage());
}
%>

```

```

<html lang="en">

<head>
<meta name="viewport" content="width=device-width, initial-scale=1">
<title>Deferred Report</title>

<link rel="stylesheet"
      href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
<link rel="stylesheet" href="css/nav-css.css">
<link rel="stylesheet" href="css/sales-portal.css">

<script
      src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min.js"></script>
<script
      src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>

<script type="text/javascript">
    function clearFrame() {
        document.getElementById("theFrame").className = "noframe";
        document.getElementById("theFrameDiv").style.background = "url(https://mir-s3-cdn-
cf.behance.net/project_modules/disp/09b24e31234507.564a1d23c07b4.gif) center center no-
repeat";
    }
    function loadFrame() {
        document.getElementById("theFrame").className = "showframe";
        document.getElementById("theFrameDiv").style.background = "url(white_background.png)";
    }
</script>

<style>
th, tr {
    text-align: center;
    width: 34%;
}

td {
    width: 34%;
}

.button {
    background-color: #2496EF;
    border: none;
    color: white;
    padding: 8px 32px;
    text-align: center;
    text-decoration: none;
    display: inline-block;
    font-size: 16px;
    margin: 4px 2px;
    cursor: pointer;
    width: 80%;
    transition: all 0.2s ease-in-out;
    border-radius: 4px;
}

.button:hover {
    background-color: #79bfff;
}

table {

```

```

        border: 1px solid black;
        overflow: scroll;
        border-collapse: collapse;
        width: 100%;
    }

    tr:nth-child(even) {
        background-color: #f2f2f2;
    }

    .noframe {
        display: none;
    }

    .showframe {
        display: block;
    }

    .theFrame {
        height: 350px;
    }

    #tabdiv {
        display: block;
        height: 200px;
        overflow-y: scroll;
        overflow: auto;
    }

    form {
        width: 100%;
    }

    #header {
        text-align: center;
    }

    #complete {
        color: #4CB04E;
        font-weight: bold;
    }

    #not_complete {
        color: red;
        font-weight: bold;
    }
</style>
</head>

<body>
    <div class="container" id="mainContainer">

        <nav class="navbar navbar-default" id="page-nav">
            <div class="container-fluid">
                <div class="navbar-header">
                    <button type="button" class="navbar-toggle" data-
toggle="collapse"
                    data-target="#myNavbar">
                        <span class="icon-bar"></span> <span class="icon-
bar"></span> <span
                            class="icon-bar"></span>

```

```

        </button>
        <a class="navbar-brand" href="index.jsp">Company ABC</a>
    </div>
    <div class="collapse navbar-collapse" id="myNavbar">
        <ul class="nav navbar-nav">
            <li><a href="index.jsp">Home</a></li>
            <li><a href="regular-report.jsp">Regular Report</a></li>
            <li class="active"><a href="#">Deferred Report</a></li>
            <li><a href="reportcaster.jsp">ReportCaster</a></li>
        </ul>
        <ul class="nav navbar-nav navbar-right">
            <li><a href="Logout.jsp" id="Log-out"><span
                class="glyphicon glyphicon-log-
out"></span> Log Out</a></li>
        </ul>
    </div>
</nav>

<div id="header">
    <h2>Deferred Reports</h2>
    <p>View the status of your deferred reports and click to view
        them!</p>
    <br></div>

<table>
    <tr>
        <th>Report Name</th>
        <th>Report Status</th>
        <th>Action</th>
    </tr>
</table>

<div id=tabdiv>
    <table>
        <%
            for (int i = 0; i < reportStatus.size(); i++) {
        %>
        <tr>
            <%
                if (reportStatus.get(i).equals("COMPLETE")) {
            %>
            <td><%=reportNames.get(i)%></td>
            <td id="complete"><%=reportStatus.get(i)%></td>
            <td>
                <form name="x" action="defer-redirect.jsp"
                    onclick="clearFrame()">
                    <input type="hidden" name="x"
                    value="<%=reportNames.get(i)%>" />
                    <button class="button" type="submit">View
                    Report</button>
                </form>
            </td>
            <%
                } else {
            %>
            <td><%=reportNames.get(i)%></td>

```

```

                <td id="not_complete"><%=reportStatus.get(i)%></td>
                <td></td>
                <%
                    }
                %>
            </tr>
            <%
                }
            %>
        </table>
    </div>

    <div class="theFrame" id="theFrameDiv">
        <div>
            <iframe onLoad="loadFrame()" name="name_of_iframe" id="theFrame"
                width="100%" height="450px" style="border: none"></iframe>
        </div>
    </div>
</body>
</html>

<%
}
%>

```

**Deferred-redirect.jsp:** This file executes the REST calls once the “View Report” button is clicked and feeds the HTML response back to the iFrame of the appropriate deferred report.

```

<%@ page
    import="java.util.*,
        java.net.*,
        java.io.*,
        java.awt.Frame,
        java.io.BufferedReader,
        java.io.File,
        java.io.FileOutputStream,
        java.io.IOException,
        java.io.InputStream,
        java.io.InputStreamReader,
        java.io.PrintWriter,
        java.io.UnsupportedEncodingException,
        java.nio.charset.StandardCharsets,
        java.util.ArrayList,
        org.apache.http.Header,
        org.apache.http.HttpResponse,
        org.apache.http.NameValuePair,
        org.apache.http.client.*,
        org.apache.http.client.entity.UrlEncodedFormEntity,
        org.apache.http.client.methods.*,
        org.apache.http.impl.client.CloseableHttpClient,
        org.apache.http.impl.client.HttpClients,
        org.apache.http.message.BasicNameValuePair,
        org.apache.commons.io.IOUtils"%>

<%
/**
 * Defines hostname, cookies[], client, pulled from logincheck.jsp to maintain the session

```



```

* Defines reportTitle which is pulled from the form block in deferred.jsp
*/
String hostname = (String) session.getAttribute("hostname");
Header[] cookies = (Header[]) session.getAttribute("cookies");
CloseableHttpClient client = (CloseableHttpClient) session.getAttribute("client");
String reportTitle = request.getParameter("x");
String ticket = "";

/**
 * Gets the url for all deferred report tickets
 */
String listTicketsURL = "http://" + hostname
    + ".ibi.com:8080/ibi_apps/rs?IBIRS_action=listTickets&IBIRS_service=defer";

/**
 * getTickets is defined to send the get request using listTicketsURL
 */
HttpGet getTickets = new HttpGet(listTicketsURL);

/**
 * For loop attaches the cookies to the http get request (getTickets)
 */
for (int h = 0; h < cookies.length; h++) {
    getTickets.addHeader(cookies[h].getName(), cookies[h].getValue());
}

/**
 * Defines HttpResponse outside the try block
 */
HttpResponse httpResponse = null;

/**
 * httpResponse is set to the executed getTickets request
 * Gets status code for possible error message
 * Catch makes sure to prevent executing any errors
 */
try {
    httpResponse = client.execute(getTickets);
    int statusCode = httpResponse.getStatusLine().getStatusCode();
} catch (Exception ex) {
    System.out.println("HttpResponse Error: " + ex.getMessage());
}

/**
 * Creates and defines rstream and br to read a webpage
 */
try {
    InputStream rstream = httpResponse.getEntity().getContent();
    BufferedReader br = new BufferedReader(new InputStreamReader(rstream));
    String line;
    String newOutput = "";

    /**
     * While loop reads the page line by line and sets it to a "newOutput" string
     */
    while ((line = br.readLine()) != null) {
        newOutput = newOutput + line;
    }

    /**
     * newOutput string is split up by "<" and put into "stringArray" array

```

```

        * Each line in "stringArray" is checked to see if it contains "description="
+ reportTitle & "item _jt="IBFSMRDefTicketObject""
        * if it does, then that is split up by spaces and pulls out the ticket for
that particular deferred report
        * closes the buffered reader
    */
    String[] stringArray = newOutput.split("<");
    for (int i = 0; i < stringArray.length; i++) {
        if (stringArray[i].contains("description=\"" + reportTitle)
            && stringArray[i].contains("item
_jt=\"IBFSMRDefTicketObject\"")) {
            String[] stringArray2 = stringArray[i].split("\\s+");
            for (int j = 0; j < stringArray2.length; j++) {
                if (stringArray2[j].contains("name=")) {
                    ticket = stringArray2[j].replace("name=\"", "");
                    ticket = ticket.replace("\"", "");
                }
            }
        }
    }
    br.close();
} catch (Exception ex) {
    System.out.println("RStream Error: " + ex.getMessage());
}

/**
 * Creates the url for getting a deferred report using the newly received "ticket"
 */
String deferredReportURL = "http://" + hostname
    + ".ibi.com:8080/ibi_apps/rs?IBIRS_action=getReport&IBIRS_ticketName=" +
ticket
    + "&IBIRS_service=defer";

/**
 * Executes the get request using the "deferredReportURL"
 */
HttpGet getReport = new HttpGet(deferredReportURL);

/**
 * For loop attaches cookies to the "getReport" get request
 */
for (int h = 0; h < cookies.length; h++) {
    getReport.addHeader(cookies[h].getName(), cookies[h].getValue());
}

/**
 * Defines httpResponse2 and replacedString outside the try block
 */
HttpResponse httpResponse2 = null;
String replacedString = "";

/**
 * Executes the "getReport" get request
 * Gets status code for possible error
 * Gets the response for the request
 */
try {
    httpResponse2 = client.execute(getReport);
    int statusCode = httpResponse2.getStatusLine().getStatusCode();
    String result = IOUtils.toString(httpResponse2.getEntity().getContent(),
StandardCharsets.UTF_8);

```

```
        replacedString = result.replace("/ibi_apps/", "http://" + hostname +  
".ibi.com:8080/ibi_apps/");  
    } catch (Exception ex) {  
        System.out.println("HttpResponse2 Error: " + ex.getMessage());  
    }  
  
    /**  
     * Sends the deferred report to the iFrame in deferred.jsp  
     */  
%>  
  
<%=replacedString%>
```

# ReportCaster

Company ABC

Home

Regular Report

Deferred Report

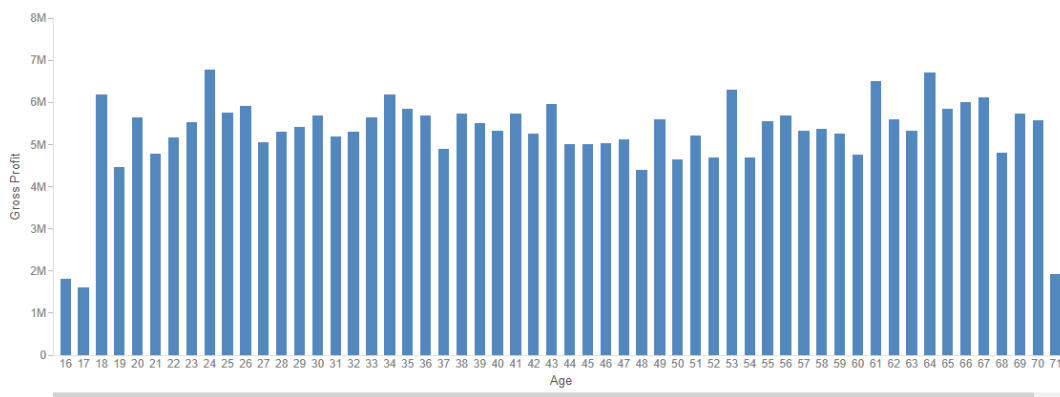
ReportCaster

Log Out

## ReportCaster

Click to view or delete your reports!

Report Name	To Email	From Email	Start Time	End Time	Delete
Manager Hours Report	to@email.com	from@email.com	02-13-2028 19:40:00 EST		
Delayed Shipments Report	to@email.com	from@email.com	08-13-2019 18:59:00 EDT		
Gross Profit Chart	to@email.com	from@email.com	09-13-2020 19:38:00 EDT	12-31-2099 23:59:00 EST	
Cost of Products Chart	to@email.com	from@email.com	08-13-2019 18:44:00 EDT	12-31-2019 23:59:00 EST	



The user can select from a number of reports in the Public folder in the WebFOCUS Repository, and view their schedules. These report names are listed in a table with their schedule's To and From emails, their start and end times, and an option to delete them. The report request is executed via its corresponding REST call as well as the reportcaster.jsp, rc-redirect.jsp, delete-redirect.jsp files (as shown below):

## REST Call and URL Format

The REST call made to GET the contents of the Public Folder in the WebFOCUS Repository is as follows:

**HTTP Method:** GET

**REST URL Format:** [http://host:port/ibi\\_apps/rs/ibfs/wfc/Repository/FolderName](http://host:port/ibi_apps/rs/ibfs/wfc/Repository/FolderName)

where:

**host**

Is the name of the system where WebFOCUS is installed.

**port**

Is the port number used by WebFOCUS.

**FolderName**

Is the name of the folder used for the stored WebFOCUS report. If the folder used for the stored WebFOCUS report exists as a subfolder, then the path to the subfolder name must be included in the REST URL. For example, TopFolderName/SubFolderName.

**Body Format:**

IBIRS\_action=get

The REST call made to GET the specified scheduled report is as follows:

**HTTP Method:** GET

**REST URL Format:**

[http://host:port/ibi\\_apps/rs/ibfs/wfc/Repository/FolderName/scheduleName](http://host:port/ibi_apps/rs/ibfs/wfc/Repository/FolderName/scheduleName)

where:

**host**

Is the name of the system where WebFOCUS is installed.

**port**

Is the port number used by WebFOCUS.

**FolderName**

Is the name of the folder used for the stored WebFOCUS report. If the folder used for the stored WebFOCUS report exists as a subfolder, then the path to the subfolder name must be included in the REST URL. For example, TopFolderName/SubFolderName.

**scheduleName**

Is the name of the selected WebFOCUS scheduled report.

**Body Format:**

IBIRS\_action=get

The REST call made to DELETE the specified scheduled report is as follows:

**HTTP Method:** DELETE

**REST URL Format:**

[http://host:port/ibi\\_apps/rs/ibfs/wfc/Repository/FolderName/scheduleName](http://host:port/ibi_apps/rs/ibfs/wfc/Repository/FolderName/scheduleName)

where:

**host**

Is the name of the system where WebFOCUS is installed.

**port**

Is the port number used by WebFOCUS.

**FolderName**

Is the name of the folder used for the stored WebFOCUS report. If the folder used for the stored WebFOCUS report exists as a subfolder, then the path to the subfolder name must be included in the REST URL. For example, TopFolderName/SubFolderName.

**scheduleName**

Is the name of the selected WebFOCUS scheduled report.

## Code & JSP Files

REPORTCASTER.JSP: This file displays the webpage that has a table displaying schedules with their To and From emails, their start and end times, a delete option, and their corresponding report.

```
<!DOCTYPE html>
```

```
<%@ page
import="java.util.*,
    java.net.*,
    java.io.*,
    java.text.*,
    java.awt.Frame,
    java.io.BufferedReader,
    java.io.File,
    java.io.FileOutputStream,
    java.io.IOException,
    java.io.InputStream,
    java.io.InputStreamReader,
    java.io.PrintWriter,
    java.io.UnsupportedEncodingException,
```

```

        java.util.ArrayList,
        org.apache.http.Header,
        org.apache.http.HttpResponse,
        org.apache.http.NameValuePair,
        org.apache.http.client.*,
        org.apache.http.client.entity.UrlEncodedFormEntity,
        org.apache.http.client.methods.*,
        org.apache.http.impl.client.CloseableHttpClient,
        org.apache.http.impl.client.HttpClients,
        org.apache.http.message.BasicNameValuePair"%>

<%
    /**
     * Redirects the user to the login page if they don't have an active session
     */
    String user = (String) session.getAttribute("user");
    if (user == null) {
        response.sendRedirect("login.jsp");
    } else {
%>

<%
    /**
     * Variable hostname is defined and then pulled from the user's computer
     * Variable addr is defined and then pulled from the user's computer
     * Exceptions are caught if the program can't access the users hostname
     */
    String hostname = "Unknown";

    try {
        InetAddress addr;
        addr = InetAddress.getLocalHost();
        hostname = addr.getHostName();
    } catch (UnknownHostException ex) {
        System.out.println("Hostname cannot be resolved");
    }

    /**
     * This is done in order to maintain a session
     * The cookies array stores the cookies so they can be maintained on other
webpages
     * A client is also opened in order to maintain the session
     */
    Header[] cookies = null;
    CloseableHttpClient client = HttpClients.createDefault();

    /**
     * String url is constructed using the hostname
     */
    String url = "http://" + hostname + ".ibi.com:8080/ibi_apps/rs/ibfs";
    HttpPost method = new HttpPost(url);

    /**
the session
     * NameValuePair[] is made in order to save the users credentials and maintain
     */
    ArrayList<NameValuePair> parameters = new ArrayList<NameValuePair>();
    parameters.add(new BasicNameValuePair("IBIRS_action", "signOn"));
    parameters.add(new BasicNameValuePair("IBIRS_userName", "admin"));
    parameters.add(new BasicNameValuePair("IBIRS_password", "admin"));

```

```

/**
 * This try catch is there to catch an error for a missing parameter saved in
the above arraylist
 */
try {
    method.setEntity(new UrlEncodedFormEntity(parameters));
} catch (Exception ex) {
    System.out.println("Set Parameters Error: "+ex.getMessage());
}

/**
 * Sets httpResponse equal to the post method
 * Saves the cookies for use on other pages
 * Sets session attributes for use on other pages
 */
try {
    HttpResponse httpResponse = client.execute(method);
    int statusCode = httpResponse.getStatusLine().getStatusCode();

    cookies = method.getHeaders("Set-Cookie");
    cookies = httpResponse.getHeaders("Set-Cookie");

    session.setAttribute("cookies", cookies);
    session.setAttribute("client", client);
    session.setAttribute("user", "IBIRS_userName");
    session.setAttribute("hostname", hostname);
} catch (Exception ex) {
    System.out.println("HttpResponse: "+ex.getMessage());
}

//=====
=====

/**
 * Creates the folderURL using the user's hostname
 */
String folderURL = "http://" + hostname
+
".ibi.com:8080/ibi_apps/rs/ibfs/WFC/Repository/Public?IBIRS_action=get";
/**
 * Defines getFolderXML as get request using the folderURL
 */

HttpGet getFolderXML = new HttpGet(folderURL);

/**
 * For loop attaches cookies to the "getFolderXML" get request
 */
for (int h = 0; h < cookies.length; h++) {
    getFolderXML.addHeader(cookies[h].getName(), cookies[h].getValue());
}

/**
 * Defines httpResponse outside the try block so it can be called outside of it
 */
HttpResponse httpResponse = null;

/**
 * Executes the get request
 * Gets status code for possible error

```

```

    * Catch displays possible error message
    */
    try {
        httpResponse = client.execute(getFolderXML);
        int statusCode = httpResponse.getStatusLine().getStatusCode();
    } catch (Exception ex) {
        System.out.println("HttpResponse Error: " + ex.getMessage());
    }

    /**
     * Creates ArrayLists for the report names, report titles, to emails, from emails,
start times, end times, and rest urls
     */
    List<String> reportNames = new ArrayList<String>();
    List<String> reportTitles = new ArrayList<String>();
    List<String> toEmails = new ArrayList<String>();
    List<String> fromEmails = new ArrayList<String>();
    List<String> startTimes = new ArrayList<String>();
    List<String> endTimes = new ArrayList<String>();
    List<String> restUrls = new ArrayList<String>();

    /**
     * Creates rstream and br to read the XML
     */
    try {
        InputStream rstream = httpResponse.getEntity().getContent();
        BufferedReader br = new BufferedReader(new InputStreamReader(rstream));

        /**
         * Defines string line and newOutput
         */
        String line;
        String newOutput = null;
        /**
         * While loop reads through the XML line by line
         */
        while ((line = br.readLine()) != null) {
            newOutput = line;
            /**
             * The lines are split up by spaces into stringArray[]
             * For loop goes through the stringArray to see if one of the strings
contains "rsPath=" and "sch"
             * Gets the path and adds it to a complete url and adds that url to
restUrls
             */
            String[] stringArray = line.split("\\s+");
            for (int i = 0; i < stringArray.length; i++) {
                if (stringArray[i].contains("rsPath=") &&
stringArray[i].contains("sch")) {
                    String temp = stringArray[i].replace("rsPath=\"", "");
                    String temp2 = temp.replace("\"", "?IBIRS_action=get");
                    String sub = "http://" + hostname + ".ibi.com:8080" +
temp2;
                    restUrls.add(sub);
                }
            }
            /**
             * Splits up XML by "<" into samp[]
             * Checks to each in samp[] to see if it contains "description=" &
"typeDescription=\"Schedule\"
             * If so, it is saved to descLine

```



```

        * descLine is the split up by quote space (" ), into samp2[]
        * Each string in samp2[] is checked if it contains "description"
        * The contents of the description tag are saved to "reportTitles[]"
        */
String[] samp = line.split("<");
for (int i = 0; i < samp.length; i++) {
    if (samp[i].contains("description=\"")
        &&
samp[i].contains("typeDescription=\"Schedule\"")) {
        String descLine = samp[i];
        String[] samp2 = descLine.split("\\s+");
        for (int j = 0; j < samp2.length; j++) {
            String descLine2 = samp2[j];
            if (descLine2.contains("description")) {
                String descLine3 =
descLine2.replace("description=\"", "").replace("\\\"", "");
                reportTitles.add(descLine3);
            }
        }
        /**
        * If descLine 2 has "name=" the it is saved to
reportNames
        */
        if (descLine2.contains("name=\"")) {
            String nameLines =
descLine2.replace("name=\"", "").replace("\\\"", "");
            reportNames.add(nameLines);
        }
    }
}
}
/**
 * The buffered reader is closed
 * Catch displays possible error message
 */
br.close();
} catch (Exception ex) {
    System.out.println("RStream Error: " + ex.getMessage());
}

//For Displaying Information About Schedules-----
-----
for (int i = 0; i < restUrls.size(); i++) {
    String tempRestURL = restUrls.get(i);

    /**
     * Executes all HttpGet requests in restUrls[]
     */
    HttpGet getReport = new HttpGet(tempRestURL);

    /**
     * Defines httpResponse2 outside of the try block
     */
    HttpResponse httpResponse2 = null;

    /**
     * Executes the get request
     * Gets status code for possible error
     * Catch displays possible error message
     */
    try {

```

```

        httpResponse2 = client.execute(getReport);
        int statusCode = httpResponse2.getStatusLine().getStatusCode();
    } catch (Exception ex) {
        System.out.println("HttpResponse2 Error: " + ex.getMessage());
    }

    /**
     * Creates rstream and br to read the XML
     */
    try {
        InputStream rstream = httpResponse2.getEntity().getContent();
        BufferedReader br = new BufferedReader(new InputStreamReader(rstream));
        /**
         * Defines string line outside the while loop for use outside of it
         */
        String line;
        /**
         * Splits the line containing the XML response by spaces into
stringArray[]

         * Checks each string in stringArray[] for "destinationAddress="
         * Saves destination address in the toEmails[]
         */
        while ((line = br.readLine()) != null) {
            String[] stringArray = line.split("\\s+");
            for (int j = 0; j < stringArray.length; j++) {
                if (stringArray[j].contains("destinationAddress=")) {
                    String temp =
stringArray[j].replace("destinationAddress=\"", "");
                    String temp2 = temp.replace("\"", "");
                    toEmails.add(temp2);
                }
            }
            /**
             * Checks each string in stringArray[] for
"mailReplyAddress="

             * Saves destination address in the fromEmails[]
             */
            if (stringArray[j].contains("mailReplyAddress=")) {
                String temp =
stringArray[j].replace("mailReplyAddress=\"", "");
                String temp2 = temp.replace("\"", "");
                fromEmails.add(temp2);
            }
        }
        /**
         * Splits up "line" by "<" and saves it toStringArray2[]
         */
        String[] stringArray2 = line.split("<");
        for (int j = 0; j < stringArray2.length; j++) {
            /**
             * Checks each string in stringArray2[] if it contains
"startTime"

             * If so, it replaces the beginning and end of it with
nothing, leaving behind the unix time

             * It is then parsed into the gregorian calendar format
             * Then it is saved to startTimes[]
             */
            if (stringArray2[j].contains("startTime")) {
                String temp = stringArray2[j].replace("startTime
                _jt=\"gregCalendar\" time=\"", "");
                String temp2 = temp.replace("\\"
                timeZone=\"America/New_York\"/>", "");
            }
        }
    }
}

```

```

        long temp3 = Long.parseLong(temp2);
        Date date = new Date(temp3);
        SimpleDateFormat jdf = new SimpleDateFormat("MM-
dd-yyyy HH:mm:ss z");

        String java_date = jdf.format(date);
        startTimes.add(java_date);
    }
    /**
     * Checks each string in stringArray2[] if it contains
     * If so, it replaces the beining and end of it with
     * nothing, leaving behind the unix time
     * It is then parsed into the gregorian calendar format
     * Then it is saved to endTimes[]
     */
    if (stringArray2[j].contains("endTime")) {
        String temp = stringArray2[j].replace("endTime
        _jt=\"gregCalendar\" time=\"\", \"");

        String splitBySpace = temp;
        String[] split = temp.split("\\s+");
        String temp2 = temp.replace("\
        timeZone=\"America/New_York\"/>", "");

        long temp3 = Long.parseLong(temp2);
        Date date = new Date(temp3);
        SimpleDateFormat jdf = new SimpleDateFormat("MM-
dd-yyyy HH:mm:ss z");

        String java_date = jdf.format(date);
        endTimes.add(java_date);
    }
}

/**
 * If report does not have an endtime (aka it is only scheduled
to be sent out once) then add a blank string
 */
if (!line.contains("endTime")) {
    endTimes.add("");
}
}
/**
 * The buffered reader is then closed
 * Catch displays possible error message
 */
br.close();
} catch (Exception ex) {
    System.out.println("RStream Error: " + ex.getMessage());
}
}

%>

<html lang="en">

<head>
<title>ReportCaster</title>

<link rel="stylesheet"
    href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
<link rel="stylesheet" href="css/nav-css.css">
<script
    src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min.js"></script>
<script

```

```

src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>

<script type="text/javascript">
    function clearFrame() {
        document.getElementById("theFrame").className = "noframe";
        document.getElementById("theFrameDiv").style.background = "url(https://mir-s3-cdn-
cf.behance.net/project_modules/disp/09b24e31234507.564a1d23c07b4.gif) center center no-
repeat";
    }
    function loadFrame() {
        document.getElementById("theFrame").className = "showframe";
        document.getElementById("theFrameDiv").style.background = "url(white_background.png)";
    }
</script>

<style>
.button {
    background-color: rgba(0, 0, 0, 0);
    color: #2496EF;
    border: none;
    cursor: pointer;
    transition: all 0.2s ease-in-out;
    font-weight: 600;
}

.button:hover {
    color: #79bfff;
}

.noframe {
    display: none;
}

.showframe {
    display: block;
}

.theFrame {
    height: 350px;
    border-style:
}

table {
    border: 1px solid black;
    overflow: scroll;
    border-collapse: collapse;
    width: 100%;
}

#tabdiv {
    display: block;
    height: 200px;
    overflow-y: scroll;
    overflow: auto;
}

td {
    width: 15%;
    border:;
}

```

```

th, tr {
    text-align: center;
}

th {
    width: 15%;
}

iframe {
    margin: 10px
}

tr:nth-child(even) {
    background-color: #f2f2f2;
}

#header {
    text-align: center;
}
</style>
</head>

<body>
    <div class="container" id="mainContainer">

        <nav class="navbar navbar-default" id="page-nav">
            <div class="container-fluid">
                <div class="navbar-header">
                    <button type="button" class="navbar-toggle" data-
toggle="collapse"
                    data-target="#myNavbar">
                        <span class="icon-bar"></span> <span class="icon-
bar"></span> <span
                            class="icon-bar"></span>
                    </button>
                    <a class="navbar-brand" href="index.jsp">Company ABC</a>
                </div>
                <div class="collapse navbar-collapse" id="myNavbar">
                    <ul class="nav navbar-nav">
                        <li><a href="index.jsp">Home</a></li>
                        <li><a href="regular-report.jsp">Regular Report</a></li>
                        <li><a href="deferred.jsp">Deferred Report</a></li>
                        <li class="active"><a href="#">ReportCaster</a></li>
                        <li><a href="documentation.jsp">Documentation</a></li>
                    </ul>
                    <ul class="nav navbar-nav navbar-right">
                        <li><a href="logout.jsp" id="Log-out"><span
                            class="glyphicon glyphicon-log-
out"></span> Log Out</a></li>
                    </ul>
                </div>
            </div>
        </nav>

        <div id="header">
            <h2>ReportCaster</h2>
            <p>Click to view or delete your scheduled reports!</p>
            <br></div>

        <table>

```

```

        <tr>
            <th>Report Name</th>
            <th>To Email</th>
            <th>From Email</th>
            <th>Start Time</th>
            <th>End Time</th>
            <th>Delete</th>
        </tr>
    </table>

    <div id=tabdiv>
        <table>
            <%
                for (int i = 0; i < reportNames.size(); i++) {
            %>
            <tr>
                <td>
                    <form name="x" action="rc-redirect.jsp"
target="name_of_iframe"
                    onclick="clearFrame()">
                        <input type="hidden" name="x"
value="<%=reportNames.get(i)%>"></input>
                        <button class="button"
type="submit"><%=reportTitles.get(i)%></button>
                    </form>
                </td>
                <td><%=toEmails.get(i)%></td>
                <td><%=fromEmails.get(i)%></td>
                <td><%=startTimes.get(i)%></td>
                <td><%=endTimes.get(i)%></td>
                <td id="delete_icon">
                    <form action="delete-redirect.jsp">
                        <input type="hidden" name="x"
value="<%=reportNames.get(i)%>" />
                        <input type="image"
src="https://image.flaticon.com/icons/svg/69/69324.svg"
height=20px width=20px>
                    </form>
                </td>
            </tr>
            <%
                }
            %>
        </table>
    </div>

    <div class="theFrame" id="theFrameDiv">
        <iframe onLoad="loadFrame()" name="name_of_iframe" id="theFrame"
width="100%" height="450px" style="border-style: none;"></iframe>
    </div>

</div>
</body>
</html>

<%
}
%>

```

RC-REDIRECT.JSP: This file executes the REST calls once the hyperlinked report name is clicked and feeds the HTML response back to the iFrame.

```
<%@ page import="java.util.*,
    java.net.*,
    java.io.*,
    java.awt.Frame,
    java.io.BufferedReader,
    java.io.File,
    java.io.FileOutputStream,
    java.io.IOException,
    java.io.InputStream,
    java.io.InputStreamReader,
    java.io.PrintWriter,
    java.io.UnsupportedEncodingException,
    java.nio.charset.StandardCharsets,
    java.util.ArrayList,
    org.apache.http.Header,
    org.apache.http.HttpResponse,
    org.apache.http.NameValuePair,
    org.apache.http.client.*,
    org.apache.http.client.entity.UrlEncodedFormEntity,
    org.apache.http.client.methods.*,
    org.apache.http.impl.client.CloseableHttpClient,
    org.apache.http.impl.client.HttpClients,
    org.apache.http.message.BasicNameValuePair,
    org.apache.commons.io.IOUtils" %>

<%
    /**
     * Defines hostname and sets it equal to attribute "hostname" from logincheck.jsp
     * cookies are defined and filled with "cookies" pulled from logincheck.jsp
     * client is defined and set equal to attribute "client" pulled from logincheck.jsp
     * scheduleName is defined and pulled from the "x" parameter when a user selects a
report to view in reportcaster.jsp
     */
    String hostname = (String) session.getAttribute("hostname");
    Header[] cookies = (Header[]) session.getAttribute("cookies");
    CloseableHttpClient client = (CloseableHttpClient) session.getAttribute("client");
    String scheduleName = request.getParameter("x");
    String reportURL = "";

    /**
     * Defines scheduleURL and sets it to a constructed url by using hostname and
scheduleName
     */
    String scheduleURL = "http://" + hostname +
".ibi.com:8080/ibi_apps/rs/ibfs/WFC/Repository/Public/" + scheduleName + "?IBIRS_action=get";

    /**
     * Defines getXML and sets it equal to get request using scheduleURL
     */
    HttpGet getXML = new HttpGet(scheduleURL);

    /**
     * Creates a for loop to attach cookies to the getXML get request to access the reports
     */
    for (int h = 0; h < cookies.length; h++) {
        getXML.addHeader(cookies[h].getName(), cookies[h].getValue());
    }
}
```

```

}

HttpResponse httpResponse = null;

/**
 * Sets the newly defined httpResponse to execute the getXML response
 */
try {
    httpResponse = client.execute(getXML);
    int statusCode = httpResponse.getStatusLine().getStatusCode();
} catch (Exception ex) {
    System.out.println("HttpResponse Error: " + ex.getMessage());
}

/**
 * Gets the XML response and reads through it line by line
 */
try {
    InputStream rstream = httpResponse.getEntity().getContent();

    BufferedReader br = new BufferedReader(new InputStreamReader(rstream));
    String line;
    String newOutput = "";

    /**
     * While loop puts the lines into the newOutputString
     */
    while ((line = br.readLine()) != null) {
        newOutput = newOutput + line;
    }

    /**
     * newOutput is split up by spaces and saved to the stringArray array
     * For loop is created to check for a line that has "procedureName" & ".fex" (this line
will contain the report name)
     * temp is set equal to the new complete URL
     * The buffered reader is now closed
     */
    String[] stringArray = newOutput.split("\\s+");
    for (int i = 0; i < stringArray.length; i++) {
        if (stringArray[i].contains("procedureName") &&
stringArray[i].contains(".fex")) {
            String temp = stringArray[i].replace("procedureName=\"IBFS:/", "http://" +
hostname + ".ibi.com:8080/ibi_apps/rs?IBIRS_path=");
            reportURL = temp.replace("\\"", "&IBIRS_action=run&IBIRS_service=ibfs");
        }
    }
    br.close();
} catch (Exception ex) {
    System.out.println("RStream Error: " + ex.getMessage());
}

/**
 * getReport is set equal to the getRequest
 */
HttpGet getReport = new HttpGet(reportURL);

/**
 * For loop attaches the cookies to the get request
 */
for (int h = 0; h < cookies.length; h++) {

```



```

        getReport.addHeader(cookies[h].getName(), cookies[h].getValue());
    }

    HttpResponse httpResponse2 = null;
    String replacedString = "";

    /**
     * The getReport request is executed
     * Gets status code for possible error
     * Gets the response for the request
     */
    try {
        httpResponse2 = client.execute(getReport);
        int statusCode = httpResponse2.getStatusLine().getStatusCode();
        String result = IOUtils.toString(httpResponse2.getEntity().getContent(),
StandardCharsets.UTF_8);
        replacedString = result.replace("/ibi_apps/", "http://" + hostname +
".ibi.com:8080/ibi_apps/");
    } catch (Exception ex) {
        System.out.println("HttpResponse2 Error: " + ex.getMessage());
    }

    /**
     * The proper report is sent to the iFrame on the ReportCaster page
     */
%>

<%= replacedString %>

```

**DELETE-REDIRECT.JSP:** This file executes the rest call to delete the specified scheduled report.

```

<%@ page import="java.util.*,
    java.net.*,
    java.io.*,
    java.awt.Frame,
    java.io.BufferedReader,
    java.io.File,
    java.io.FileOutputStream,
    java.io.IOException,
    java.io.InputStream,
    java.io.InputStreamReader,
    java.io.PrintWriter,
    java.io.UnsupportedEncodingException,
    java.util.ArrayList,
    org.apache.http.Header,
    org.apache.http.HttpResponse,
    org.apache.http.NameValuePair,
    org.apache.http.client.*,
    org.apache.http.client.entity.UrlEncodedFormEntity,
    org.apache.http.client.methods.*,
    org.apache.http.impl.client.CloseableHttpClient,
    org.apache.http.impl.client.HttpClients,
    org.apache.http.message.BasicNameValuePair" %>

<%
    /**
     * String hostname is defined and set equal to attribute pulled from logincheck.jsp
     * Cookies are saved into an array, which are pulled from logincheck.jsp
     (session.getAttribute)
     * Client is pulled from logincheck.jsp (session.getAttribute)
     * getParameter x gets the specific schedule name that is being deleted

```

```

*/
String hostname = (String) session.getAttribute("hostname");
Header[] cookies = (Header[]) session.getAttribute("cookies");
CloseableHttpClient client = (CloseableHttpClient) session.getAttribute("client");
String scheduleName = request.getParameter("x");

/**
 * deleteURL is defined as a constructed url that uses the host name and the schedule
name
*/
String deleteURL = "http://" + hostname +
".ibi.com:8080/ibi_apps/rs/ibfs/WFC/Repository/Public/" + scheduleName;

/**
 * HTTP request is made to delete a particular schedule using the "deleteURL" made in
the previous line
*/
HttpDelete delete = new HttpDelete(deleteURL);

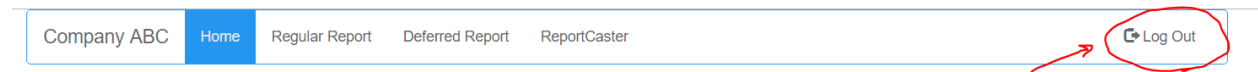
/**
 * For loop is made to delete all the cookies in the cookie array
*/
for (int h = 0; h < cookies.length; h++) {
delete.addHeader(cookies[h].getName(), cookies[h].getValue());
}

/**
 * httpResponse is defined
 * Try is made to execute a delete response
 * Catch is there to catch an error if there was no schedule to delete
*/
HttpResponse httpResponse = null;
try {
httpResponse = client.execute(delete);
int statusCode = httpResponse.getStatusLine().getStatusCode();
} catch (Exception ex) {
System.out.println("HttpResponse Error: " + ex.getMessage());
}

/**
 * This redirect updates the reportcaster page after the delete is executed
*/
response.sendRedirect("reportcaster.jsp");
%>

```

## Signing Out



The user can select the Log Out option at the top right of all signed in webpages. This will end their session and send them back to the sign-in page.

### REST Call and URL Format

The REST call made and its details to execute signing out is as follows:

**HTTP Method:** N/A

**REST URL Format:** N/A

### Code & JSP Files

LOGOUT.JSP: This file invalidates the session, and redirects the user to the login page.

```
<%@ page
import="java.util.*,
org.apache.http.Header,
org.apache.http.HttpResponse,
org.apache.http.NameValuePair,
org.apache.http.impl.client.CloseableHttpClient,
org.apache.http.impl.client.HttpClients,
org.apache.http.message.BasicNameValuePair"%>

<%
/**
 * Getting cookies to maintain the session
 * Creating a client for the session use
 *
 * Cookies are set to null to delete them and the session is invalidated
 * The user is then sent to the login page
 */
Header[] cookies = (Header[]) session.getAttribute("cookies");
CloseableHttpClient client = (CloseableHttpClient) session.getAttribute("client");

cookies = null;
session.invalidate();

response.sendRedirect("login.jsp");
%>
```

## Possible Errors

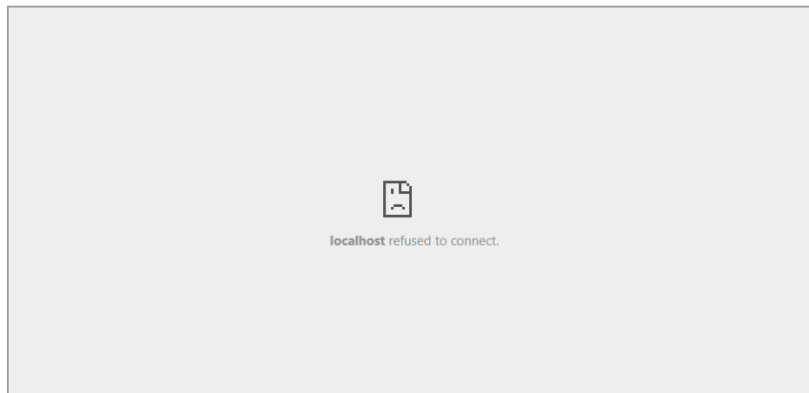
The following are some possible errors you may run into while running the web application:

- **Reporting Server Error**

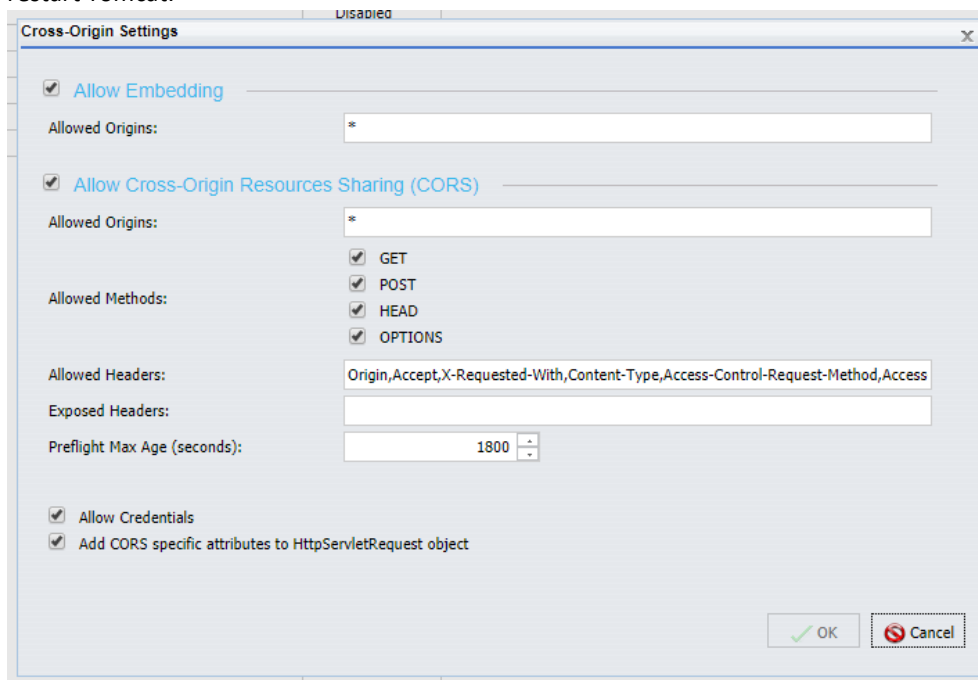
**Reporting Server Error**  
Reporting Server not running/responding Node=EDASERVE

Solution: Make sure your WebFOCUS 82 Server is started.

- **Localhost Refused to Connect (Elements are not appearing in iFrame)**



Solution: Navigate to the Administration Console. Click the **Security** tab, the **Security Zones** folder, **Default** folder, and then click **Authentication**. On the Actions bar, click **Cross-Origin Settings**. Select the **Allow Embedding** and **Allow Cross-Origin Resources Sharing (CORS)** check boxes. Save your settings and restart Tomcat.



- **EDASERVE Credentials Required Prompt**

Company ABC   Home   **Regular Report**   Deferred Report   ReportCaster   [Log Out](#)

## Available Reports

Click to view your reports!

Report Name	Action
Product Revenue Report	<a href="#">View Report</a>
Delayed Shipments Chart	<a href="#">View Report</a>
Gross Profit Chart	<a href="#">View Report</a>
Manager Hours Report	<a href="#">View Report</a>

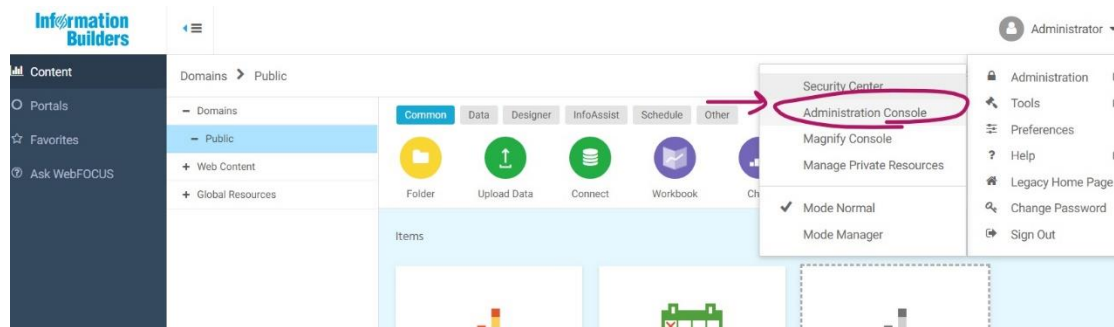
Valid credentials are required for reporting server:  
**EDASERVE**

User ID:

Password:

[Sign in](#)   [Cancel](#)

Solution: Navigate to the Administration Console.



In the **Configuration** tab, click the **Reporting Servers** folder to expand, and then click the **Server Connections** folder. Select **EDASERVE** to open up **Client Configuration**.

**WF**   **Configuration**   Security   ReportCaster   Diagnostics

**Configuration**

- Reporting Servers
  - Server Connections
    - EDASERVE**
      - Alternate Server Mapping
      - Cluster Manager
      - Legacy Cluster
      - Distribution Directories
      - Application Settings
        - Custom Settings
        - NLS Settings
        - Dynamic Language Switch
        - Redirection Settings
        - InfoAssist Properties
        - Role Update Utility
        - HTML5 Chart Extensions

**Client Configuration**

**Basic**

Node Name:  (required)

Node Description:

Host:  (required)

TCP/IP Port:  (required)

HTTP(S) Port:

Security:
 

- ☒ Prompt for Credentials
- ☐ HTTP Basic
- ☐ Kerberos
- ☐ SAP Ticket
- ☐ Service Account
- ☐ Trusted

**Advanced**

[Save](#)   [Cancel](#)

In the **Security** section, select **Service Account** and enter your server credentials. Click **Save** and refresh your web application.

#### Client Configuration

▼ Basic

Node Name:	EDASERVE	(required)
Node Description:		
Host:	JH15163	(required)
TCP/IP Port:	8120	(required)
HTTP(S) Port:	8121	
Security:	<div><input type="radio"/> Prompt for Credentials</div> <div><input type="radio"/> HTTP Basic</div> <div><input type="radio"/> Kerberos</div> <div><input type="radio"/> SAP Ticket</div> <div><input checked="" type="radio"/> Service Account</div> <div><input type="radio"/> Trusted</div>	
	User ID:	svradmin
	Password:	*****

► Advanced

Save Cancel

- **Internal Server Error**

## HTTP Status 500 – Internal Server Error

Solution: Go back to login.jsp and try signing in again.

## Additional Resources

If you would like to implement additional features to the web application, below are resources you can use for further reference:

- **WebFOCUS Embedded Business Intelligence User's Guide**  
[https://webfocusinfocenter.informationbuilders.com/wfappent/pdfs2/embedded\\_bi\\_user.pdf](https://webfocusinfocenter.informationbuilders.com/wfappent/pdfs2/embedded_bi_user.pdf)  
The WebFOCUS Embedded Business Intelligence User's Guide introduces WebFOCUS embedded business intelligence (BI) and includes comprehensive content on WebFOCUS RESTful Web Services for developers and WebFOCUS Open Portal Services, which are key components in embedded BI solutions.
- **WebFOCUS RESTful Web Service Test Page**  
[http://localhost:8080/ibi\\_apps/rs?IBIRS\\_action=TEST](http://localhost:8080/ibi_apps/rs?IBIRS_action=TEST)  
The WebFOCUS RESTful Web Service Test Page contains all of the RESTful web service calls used by WebFOCUS.