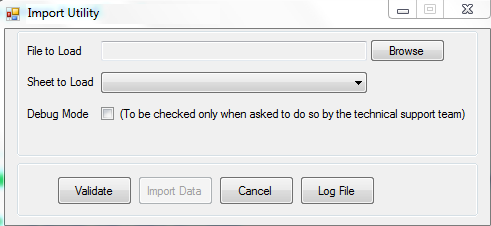
# Bulk Data Importer (BDI)

Bulk data importer is a web based application API to import bulk data into an enterprise application. This application helps the customer to import all their existing and new resources / data as and when required to their enterprise application without disturbing its performance and operations. This application service accepts only authenticated requests. It accepts the request data that to be imported in the XML format and respond back with the result in XML format. This helps 3rd party applications to interact / integrate with their enterprise application.



## Code Sample from the core service.

|  |
| --- |
| public HashMap<String, String> doImport(String xmlQueryString) { |
| StringBuffer sb = new StringBuffer(); |
| try { |
| Readproperties.readyProperty(); |
|  |
| if (xmlQueryString == null || xmlQueryString.trim().length()<=0) { |
| log.debug("Request query string is Null"); |
| responseText.put(Constants.PROCESSTYPE\_MESSAGE, "Query String is Null"); |
| return responseText; |
| } |
|  |
| if (validateQueryString(xmlQueryString)) { |
| responseText.put(Constants.PROCESSTYPE, processType); |
| CommonUtil.getZipFiles(dataZipFileName); |
| if (processType.equalsIgnoreCase(Constants.PROCESSTYPE\_VALIDATION)) { |
| sb = new DataValidator().doDataValidation(inputXMLFileName,dataZipFileName.substring(0, dataZipFileName.length() - 4)); |
| if(sb.length()>0){ |
| responseText.put(Constants.PROCESSTYPE\_STATUS, Constants.PROCESSTYPE\_MESSAGE\_FAILURE); |
| responseText.put(Constants.PROCESSTYPE\_MESSAGE, sb.toString()); |
| } else { |
| responseText.put(Constants.PROCESSTYPE\_STATUS, Constants.PROCESSTYPE\_MESSAGE\_SUCCESS); |
| responseText.put(Constants.PROCESSTYPE\_MESSAGE,""); |
| } |
| } else if (processType.equalsIgnoreCase(Constants.PROCESSTYPE\_INSERT)) { |
| log.info("inputXMLFileName>>>>>"+inputXMLFileName); |
| sb = new DataInserter().doDataInsertion(inputXMLFileName,dataZipFileName.substring(0, dataZipFileName.length() - 4)); |
| if(sb.length()>0){ |
| responseText.put(Constants.PROCESSTYPE\_STATUS, Constants.PROCESSTYPE\_MESSAGE\_FAILURE); |
| responseText.put(Constants.PROCESSTYPE\_MESSAGE, sb.toString()); |
| } else { |
| responseText.put(Constants.PROCESSTYPE\_STATUS, Constants.PROCESSTYPE\_MESSAGE\_SUCCESS); |
| responseText.put(Constants.PROCESSTYPE\_MESSAGE,""); |
| } |
| } |
| } |
|  |
| deleteDir(new File(Readproperties.probs.getProperty(Constants.KRI\_INPUT\_PROCESS\_FOLDER\_LOCATION)+dataZipFileName.substring(0, dataZipFileName.length()-4))); |
| return responseText; |
| } catch (FileNotFoundException e) { |
| log.error(e.getMessage(), e); |
| sb.append(e.getMessage()); |
| } catch (IOException e) { |
| log.error(e.getMessage(), e); |
| sb.append(e.getMessage()); |
| //removeZipFile(Readproperties.probs.getProperty(Constants.KRI\_FTP\_FOLDER\_LOCATION)+dataZipFileName); |
| deleteDir(new File(Readproperties.probs.getProperty(Constants.KRI\_INPUT\_PROCESS\_FOLDER\_LOCATION)+dataZipFileName.substring(0, dataZipFileName.length()-4))); |
| } catch (HibernateException e) { |
| log.error(e.getMessage(), e); |
| sb.append(e.getMessage()); |
| //removeZipFile(Readproperties.probs.getProperty(Constants.KRI\_FTP\_FOLDER\_LOCATION)+dataZipFileName); |
| deleteDir(new File(Readproperties.probs.getProperty(Constants.KRI\_INPUT\_PROCESS\_FOLDER\_LOCATION)+dataZipFileName.substring(0, dataZipFileName.length()-4))); |
| } catch (ParserConfigurationException e) { |
| log.error(e.getMessage(), e); |
| sb.append(e.getMessage()); |
| //removeZipFile(Readproperties.probs.getProperty(Constants.KRI\_FTP\_FOLDER\_LOCATION)+dataZipFileName); |
| deleteDir(new File(Readproperties.probs.getProperty(Constants.KRI\_INPUT\_PROCESS\_FOLDER\_LOCATION)+dataZipFileName.substring(0, dataZipFileName.length()-4))); |
| } catch (SAXException e) { |
| log.error(e.getMessage(), e); |
| sb.append(e.getMessage()); |
| //removeZipFile(Readproperties.probs.getProperty(Constants.KRI\_FTP\_FOLDER\_LOCATION)+dataZipFileName); |
| deleteDir(new File(Readproperties.probs.getProperty(Constants.KRI\_INPUT\_PROCESS\_FOLDER\_LOCATION)+dataZipFileName.substring(0, dataZipFileName.length()-4))); |
| } |
|  |
| responseText.put(Constants.PROCESSTYPE\_MESSAGE, sb.toString()); |
| return responseText; |
| } |