OpenText Sample App

# Scope Overview

OpenText Sample App is an application developed according to the specifications provided in the [accompanying document](#_Additional_Files). It is designed as a Console Application in C#. The function of the app is to monitor a pre-configured folder for existence of an XML file and PDF file. On presence of both, it processes them to send relevant mails and stamps DB audit logs.

# Tech Overview

|  |  |
| --- | --- |
| IDE | Visual Studio 2010 |
| Project Type | Console Application |
| Framework | .NET Framework 4.0 |
| Language | C# |
| Database | MS SQL Server |

# Pre-requisites

* 3 folders named as below, to be available in a pre-configured location

1. NetworkFolder
2. ErrorFolder
3. ArchiveFolder

* Scripts to be executed in MS SQL Server (attached below). They comprise of the following
  + Creation of Database, Login and User with the appropriate permissions
  + Creation of tables with appropriate constraints to hold the audit log information

# Additional Files

 



# Pre-configured values

These values are set in the application’s config file [app.config]. They can be changed by quite simply editing the config file on request by the client.

|  |  |
| --- | --- |
| FolderLocation | Pre-configured folders where the files are placed and where the app will move the files to |
| ErrorFolderLocation |
| ArchiveFolderLocation |
| WaitTime | Number of seconds which the app will wait after reading the existence of an XML file for the existence of a PDF file pair with the same name |
| XSDFolder | Folder where the Control XSD (against which the XML schema will be validated against) is placed. In the App, the Control XSD is a part of the project solution |
| EmailToBeSent | Y/N flag denoting whether Email logic is to be executed |
| DBFillingEnabled | Y/N flag denoting whether DB Audit logging logic is to be executed |
| Mail. | Mail |
| ConnectionString | SQL Connection string to connect to SQL Server and database |

# Functionality

The App is written as a Console Application. The app, when executed, searches in the pre-configured folder < FolderLocation > for existence of any XML files. If no XML file is available, it gives back the error that no XML file is available at the given location. If more than 1 XML files are available, the app takes the oldest file (based on Date Modified) for processing.

The app now waits for 60 seconds (configurable based on < WaitTime > key) before searching for a PDF file with the same filename in the same folder. If no PDF file with the same name is found, the XML file is moved to the pre-configured folder < ErrorFolderLocation > and gives back the error message that no corresponding PDF file is available.

If the XML and PDF file pair is available, the app then proceeds to validate the XML against a XSD Schema. This Schema structure is [attached](#_Additional_Files) as Control.XSD in the ‘[Additional Files](#_Additional_Files)’ section above.

If the XML fails the XSD Schema validation, the app moves the file pair to the < ErrorFolderLocation > and gives back an error stating the error and details like the line number and type of error.

Once successfully validated against the XSD schema, the app reads the XML data. The data is composed of the following structure

* Sender
* Recipients
  + Recipient List
* Subject
* Message Body

Once read, an SMTP protocol in the app delivers a mail with the information as below

* <Sender> as FROM address
* <Recipients> as TO addresses
* <Subject> as Mail Subject
* <MessageBody> as Mail Body
* PDF File as attachment

The SMTP information are configurable and are stored in the config file as < Mail. >. These can be changed as required. If SMTP information is unavailable, then a flag is available in the config file < EmailToBeSent > which can be set to not process the mailing module. The App shows a message on the status of the mailing.

The app now moves the file pair to a pre-configured folder < ArchiveFolderLocation >.

The app also provides the optional feature of logging this audit information into a database. This module execution is based on a flag < DBFillingEnabled > in the config file and can be set on/off as required based on the availability of a database system.

The scripts [attached](#_Additional_Files) should be executed in the database so that the databases, user/login are created.

2 tables are created. The primary table < AuditRecord > contains a Unique ID, Date/Time entry, Sender Address. A child table < RecipientRecord > references the Unique ID of the primary table and contains the recipient addresses.

The app inserts values to these tables on every execution of the process.

The app shows the information back to the user regarding the status of execution.

# Screenshots







