



Digital Capture and Storage of Tax Free Vouchers

Version: 0.1

Author: Rosen Rusev

|  |  |  |  |
| --- | --- | --- | --- |
| Ravi Bandaru | 05/11/2014 | Premier Tax Free | Project Manager |
| Rosen Rusev | 05/11/2014 | Premier Tax Free | Solution Architect |

Statement of Confidentiality

This document is proprietary to Premier Tax Free. It is supplied in confidence to XXX and should not be disclosed or otherwise revealed to outside parties without the express written consent of Premier Tax Free.

Document Control

Version Control

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Change control | Reason |
| 1.0 | 13/07/2014 | N/A | Initial Draft |
| 1.1 | 22/07/2014 |  |  |

Distribution

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Date | Company | Project Role |
| Chris Parkin | 13/07/2014 | Premier Tax Free | Project Sponsor |
| AJ Reekie-Mayne | 13/07/2014 | Premier Tax Free | Group Programme Manager |
| Ravi Bandaru | 13/07/2014 | Premier Tax Free | IT Manager |
| Rosen Rusev | 13/07/2014 | Premier Tax Free | Solution Architect |

Contents

[Version Control 2](#_Toc402969702)

[Version 2](#_Toc402969703)

[Date 2](#_Toc402969704)

[Change control 2](#_Toc402969705)

[Reason 2](#_Toc402969706)

[Name 2](#_Toc402969707)

[Date 2](#_Toc402969708)

[Company 2](#_Toc402969709)

[Project Role 2](#_Toc402969710)

[1. INTRODUCTION 6](#_Toc402969711)

[1.1. Project identification 6](#_Toc402969712)

[1.2. Purpose of this document 6](#_Toc402969713)

[1.3. Supporting documents 6](#_Toc402969714)

[2. OVERVIEW 6](#_Toc402969715)

[3. ARCHITECHTURE 7](#_Toc402969716)

[3.1. System Architecture 7](#_Toc402969717)

[3.2. Application Architecture 7](#_Toc402969718)

[3.3. Hardware Requirements 8](#_Toc402969719)

[3.3.1. Central Web server 8](#_Toc402969720)

[3.3.2. Database 8](#_Toc402969721)

[3.3.3. Image store 8](#_Toc402969722)

[3.3.4. Remote client 8](#_Toc402969723)

[4. CONFIGURATION 9](#_Toc402969724)

[4.1. System Configuration 9](#_Toc402969725)

[4.1.1. Central Web server 9](#_Toc402969726)

[4.1.2. Remote client 9](#_Toc402969727)

[4.1.3. Merchants site 9](#_Toc402969728)

[5. DETAILED DESIGN 10](#_Toc402969729)

[5.1.1. Screens 11](#_Toc402969730)

[5.1.1.1. Login 11](#_Toc402969731)

[5.1.1.2. Scan 11](#_Toc402969732)

[5.1.1.3. Search 12](#_Toc402969733)

[5.2. Business Services 12](#_Toc402969734)

[5.2.1. Diagram 12](#_Toc402969735)

[5.3. Data 12](#_Toc402969736)

[5.3.1. Scan Image Methods 12](#_Toc402969737)

[5.3.2. Working with folders Methods 14](#_Toc402969738)

[5.3.3. Version update Methods 15](#_Toc402969739)

[5.3.4. Transfer file Methods 16](#_Toc402969740)

[5.3.5. General Methods 16](#_Toc402969741)

[5.3.6. TRS related Methods 16](#_Toc402969742)

[5.3.7. PR related Methods 17](#_Toc402969743)

[5.3.8. Download Methods 17](#_Toc402969744)

[5.3.9. Emailing Methods 17](#_Toc402969745)

[5.3.10. Schema Objects 17](#_Toc402969746)

[5.3.10.1. VoucherInfo 17](#_Toc402969747)

[5.3.10.2. VoucherInfo2 18](#_Toc402969748)

[5.3.10.3. VoucherInfo3 18](#_Toc402969749)

[5.3.10.4. fileInfo 19](#_Toc402969750)

[5.3.10.5. file2Info 19](#_Toc402969751)

[5.3.10.6. FolderInfo 19](#_Toc402969752)

[5.3.10.7. UpdateFileInfo 20](#_Toc402969753)

[5.3.10.8. TransferFileInfo 20](#_Toc402969754)

[5.3.10.9. RetailerPrinterInfo 20](#_Toc402969755)

[5.3.10.10. IScanService interface 21](#_Toc402969756)

[5.3.11. Table Description 22](#_Toc402969757)

[5.3.11.1. Voucher table 22](#_Toc402969758)

[5.3.11.2. Folder 22](#_Toc402969759)

[5.3.11.3. File 23](#_Toc402969760)

[5.3.11.1. User Activity History 23](#_Toc402969761)

[5.3.11.2. History 24](#_Toc402969762)

[5.3.12. Security 25](#_Toc402969763)

[5.3.12.1. Central Web server 25](#_Toc402969764)

[5.3.12.2. File server 25](#_Toc402969765)

[5.3.12.3. Database 25](#_Toc402969766)

[5.3.12.4. Client 25](#_Toc402969767)

# INTRODUCTION

## Project identification

Project Name: Voucher scanning system – Implementation

Project Code: VSS001

## Purpose of this document

The purpose of this document is to provide an overall basis for the successful technical implementation of the project. This document addresses the technical solution that Premier Tax Free develop and deliver as detailed in the Requirements Document titled Scanning\_Software\_tech\_spec.docx. It aims to detail the individual technical functional units within the system and how they will be developed.

## Supporting documents

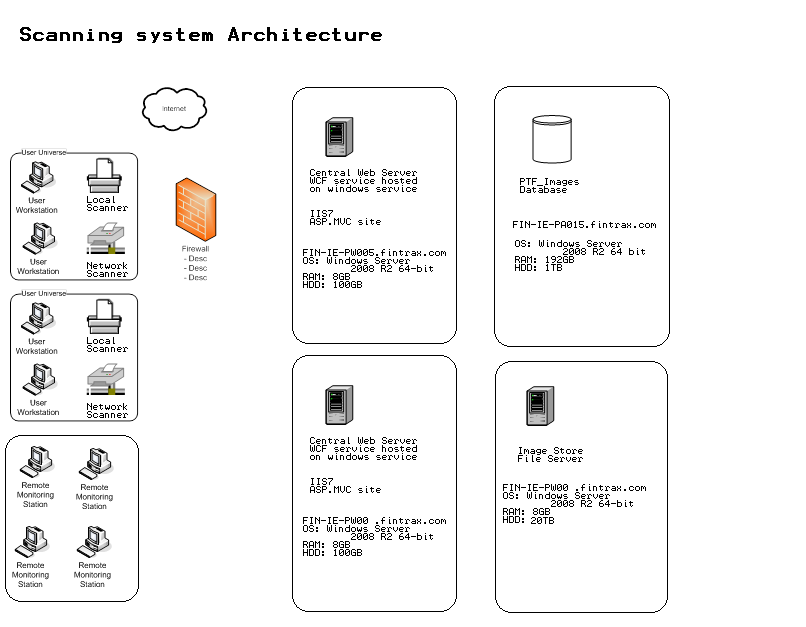
# OVERVIEW

Scanning system is a system for taking images of vouchers from any image source, maintaining the image, signing it and transporting to central server. System supports showing, downloading images. It can search by any Meta data fields taken during scan. It also provides interfaces to other system for showing images of vouchers.

# ARCHITECHTURE

## System Architecture

Scanning System contains two central servers, file server and database. Central servers run same software and are fully replicable one with the other. Database store records for the scans. It provides quick search facility to the system. File Server on its hierarchy folder structure contains all taken images.



## Application Architecture

System is built as a server – client architecture. The server is WCF service hosted on Windows service. Client is Click-Once Windows Forms application. The server operates database and images store. It provides facilities for uploading images, downloading images, updating images, deleting images and merging images and image data. There is a MVC site running on same machine. It allows selecting voucher images, VAT invoices and Meta data.

## Hardware Requirements

## Central Web server

OS: Windows Server 2008 R2

Role: Web server, IIS7

RAM: 8GB

HDD: 100GB

## Database

OS: Windows Server 2008 R2

Role: Database server

RAM: 192GB

HDD: 1TB

## Image store

OS: Windows Server 2008 R2

Role: File Server

RAM: 8GB

HDD: 20 TB

## Remote client

OS: Windows 7

Role: Workstation

RAM: 4GB

HDD: 20GB

# CONFIGURATION

## System Configuration

## Central Web server

UPLOADFOLDER – Folder WCF service uses for uploading vouchers

VOUCHERSFOLDER – Folder WCF service uses to process vouchers

VOCUHERSEXPORTFOLDER – Folder WCF service uses to export vouchers

UPLOADERRORS – Folder WCF service uses to report errors

VERSIONFOLDER – Folder central server uses to provide version

pfxFileFullPath – Certificate file path

PTFLogoFileFullPath – PTF logo file

REPORTINGSERVER\_USER – Report server user name

REPORTINGSERVER\_PASS – Report server password

REPORTINGSERVER\_DOMAIN – Report server domain

PTF\_ImagesDB – PTF\_Images database connection string

PTF\_DB – PTF connection string

## Remote client

LiveServerIP – TRS server IP address

TestServerIP – Test TRS server IP address

USE\_SCAN\_SERVER – Flag if client should use local scan server

ScanServerLocalUrl – Scan server URL

ScanServerPath – Scan server path

ScanServerIP – Scan server IP

ClearScanDirectory – Flag if application deletes temporary scan folder

## Merchants site

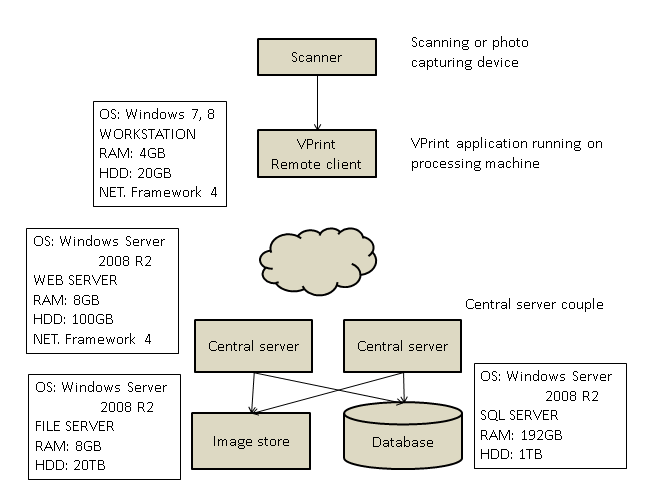
PTF\_ConnectionString – PTF connection string

PTF\_Images – PTF connection string

PTF\_ImagesConnectionString – PTF\_Images entity connection string

VOUCHERSFOLDER – Folder for image transfer

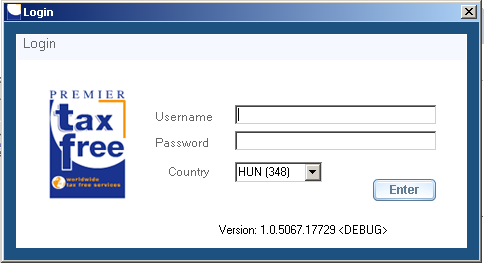
# DETAILED DESIGN

User Interface

## Screens

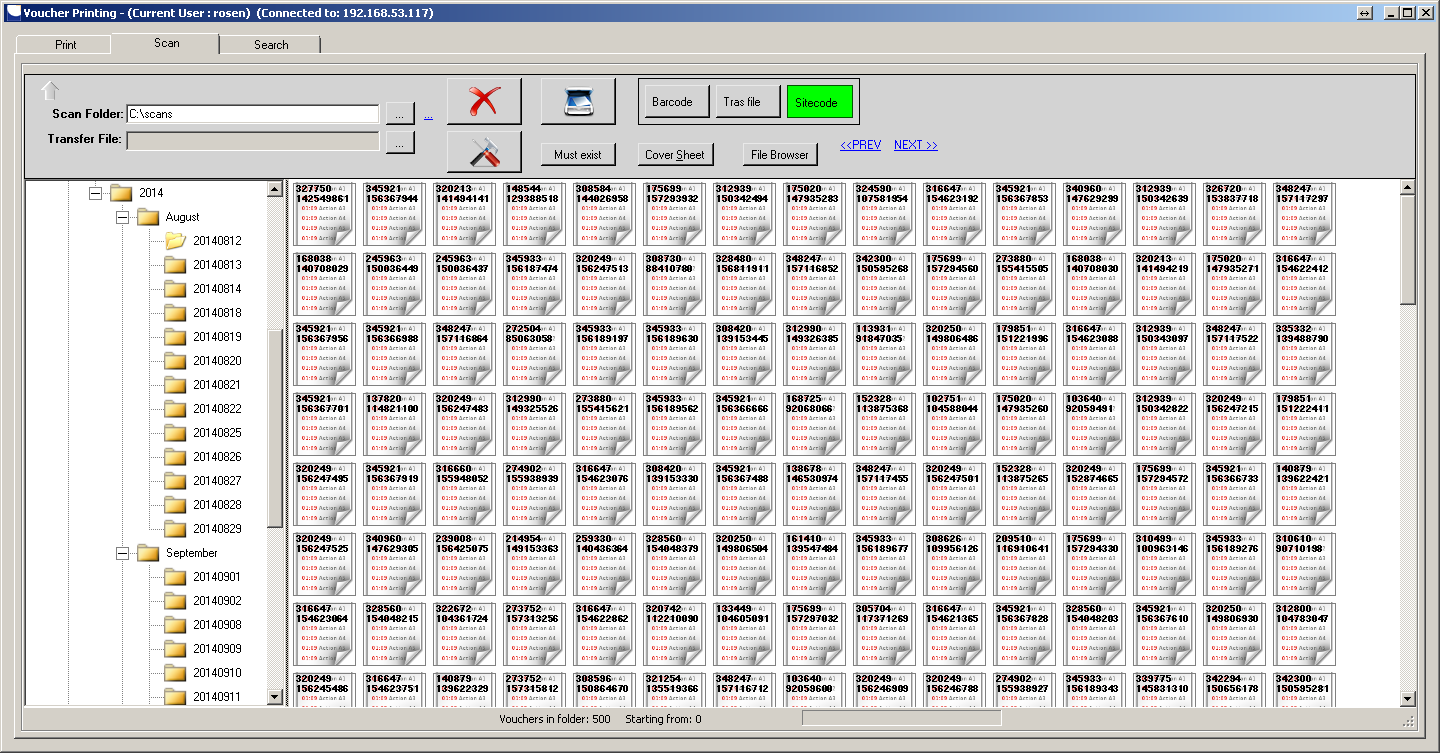
## Login

Login form is in use while user login the system



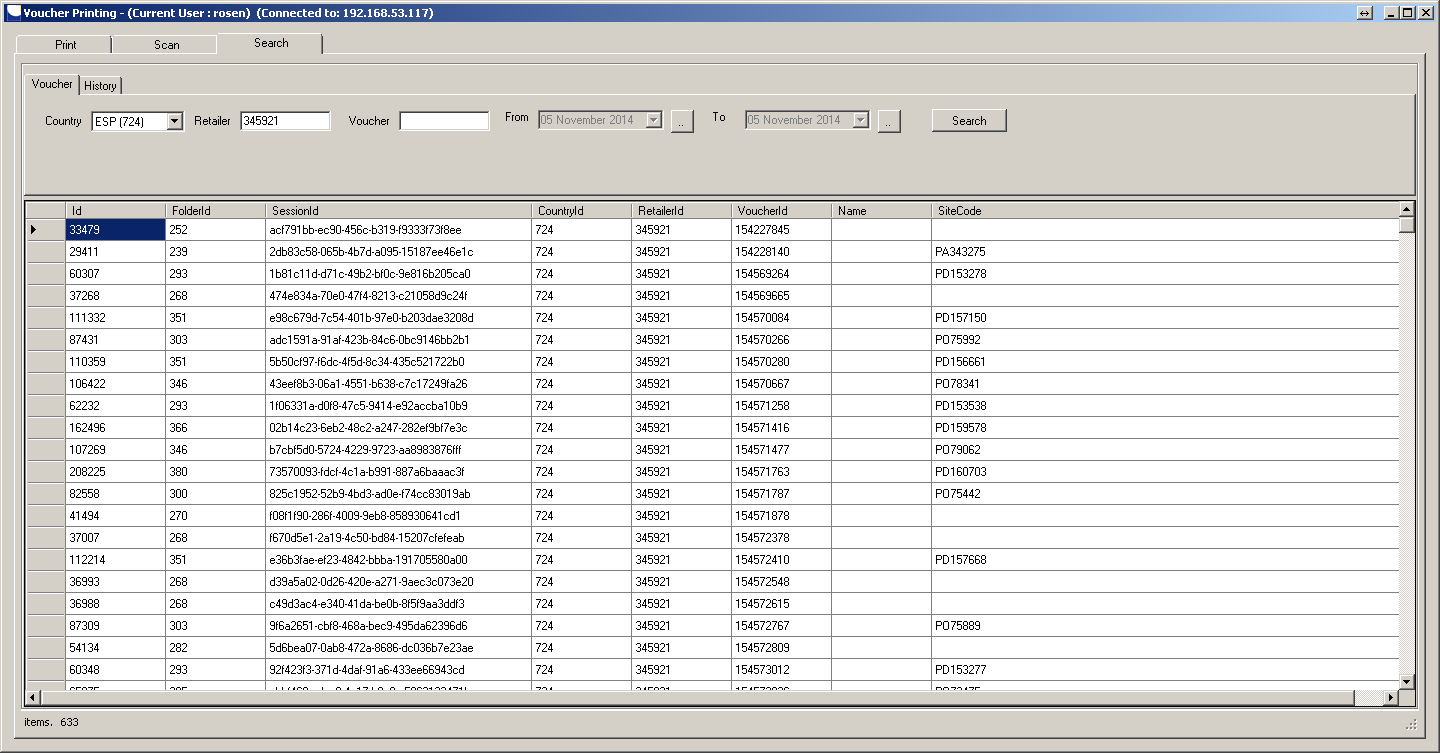
## Scan

Scan form is in use for processing voucher images



## Search

Search form is in use for searching voucher images in central image store.



## Business Services

## Diagram

## Data

## Scan Image Methods

##### Delete file by name

[OperationContract]

[FaultContract(typeof(MyApplicationFault))]

void Delete(string fileName, int countryId, int retailerId, int voucherId, string s1, string s2);

##### Read voucher info by Id

[OperationContract]

[FaultContract(typeof(MyApplicationFault))]

VoucherInfo2 ReadVoucherInfo(int Id, string copyToFolder, string s1, string s2);

##### Read data by countryId, retailerId

[OperationContract]

[FaultContract(typeof(MyApplicationFault))]

List<VoucherInfo> ReadData(int countryId, int retailerId, string s1, string s2);

##### Read data by Id

[OperationContract]

[FaultContract(typeof(MyApplicationFault))]

byte[] ReadData2(int id, bool isVoucher, int start, int length, string s1, string s2);

##### Save data synch

[OperationContract]

[FaultContract(typeof(MyApplicationFault))]

void SaveData(string serverDirName, string fileName, byte[] data, string s1, string s2);

##### Save data async

[OperationContract]

[FaultContract(typeof(MyApplicationFault))]

void SaveDataAsync(string serverDirName, string fileName, byte[] data, long position, string s1, string s2);

##### Commit voucher changes

[OperationContract]//(IsOneWay = true)]

[FaultContract(typeof(MyApplicationFault))]

void CommitVoucherChanges(string serverDirName, int jobId, int countryId, int retailerId, int voucherId, int? folderId, string siteCode, string barCode,

int locationId, int userId, string s1, string s2);

##### Commit voucher changes

[OperationContract]//(IsOneWay = true)]

[FaultContract(typeof(MyApplicationFault))]

void CommitVoucherChangesModify\_V2(string serverDirName, int jobId, int countryId, int retailerId, int voucherId, int? folderId,

string siteCode, string barCode, int locationId, int userId, int typeId, ChangeContentType action, string s1, string s2);

##### Commit file changes

[OperationContract]//(IsOneWay = true)]

[FaultContract(typeof(MyApplicationFault))]

void CommitVoucherChangesModify(string serverDirName, int jobId, int countryId, int retailerId, int voucherId, int? folderId,

string siteCode, string barCode, int locationId, int userId, ChangeContentType action, string s1, string s2);

##### Commit file changes

[OperationContract]

[FaultContract(typeof(MyApplicationFault))]

void CommitFileChanges(string serverDirName, int countryId, int? folderId,

int locationId, int userId, string s1, string s2);

##### Validate voucher by countryId, retailerId, voucherId

[OperationContract]//(IsOneWay = true)]

[FaultContract(typeof(MyApplicationFault))]

void ValidateVoucher(int countryId, bool ss, int retailerId, int voucherId, bool voucherMustExist, string s1, string s2);

##### Find voucher image by countyId, voucherId

[OperationContract]

[FaultContract(typeof(MyApplicationFault))]

string FindVoucher(int countryId, int voucherId, int voucherIdCD, string s1, string s2);

##### Find voucher image by countryId, voucherId

[OperationContract]

[FaultContract(typeof(MyApplicationFault))]

int[] FindVoucherImage(int countryId, int voucherId, int voucherIdCD, string s1, string s2);

## Working with folders Methods

##### Add folder

[OperationContract]//(Action = "\*", ReplyAction = "\*")]

[FaultContract(typeof(MyApplicationFault))]

void AddFolder(int? toParentId, string name, int countryId, int userId, string s1, string s2);

##### Update folder byfolderId

[OperationContract]//(Action = "\*", ReplyAction = "\*")]

[FaultContract(typeof(MyApplicationFault))]

void UpdateFolder(int folderId, string name, int? parentId, int countryId, int userId, string s1, string s2);

##### Delete folder by folderId

[OperationContract]//(Action = "\*", ReplyAction = "\*")]

[FaultContract(typeof(MyApplicationFault))]

void DeleteFolder(int folderId, string s1, string s2);

##### Delete file in folder by Id

[OperationContract]//(Action = "\*", ReplyAction = "\*")]

[FaultContract(typeof(MyApplicationFault))]

void DeleteFile(int id, bool isVoucher, string s1, string s2);

##### Delete all files in folder

[OperationContract]//(Action = "\*", ReplyAction = "\*")]

[FaultContract(typeof(MyApplicationFault))]

void DeleteAllFilesInFolder(int folderId, string s1, string s2);

##### Renames folder by folderId

[OperationContract]//(Action = "\*", ReplyAction = "\*")]

[FaultContract(typeof(MyApplicationFault))]

void RenameFolder(int folderId, string name, string s1, string s2);

##### Select folder by parentId

[OperationContract]//(Action = "\*", ReplyAction = "\*")]

[FaultContract(typeof(MyApplicationFault))]

List<FolderInfo> SelectFoldersByParent(int? parentId, int createdByIsoId, string s1, string s2);

##### Select files by folder

[OperationContract]//(Action = "\*", ReplyAction = "\*")]

[FaultContract(typeof(MyApplicationFault))]

List<fileInfo> SelectFilesByFolder(int folderId, string s1, string s2);

##### Select files by folder

[OperationContract]//(Action = "\*", ReplyAction = "\*")]

[FaultContract(typeof(MyApplicationFault))]

List<fileInfo> SelectFilesByFolder2(int folderId, int skip, int take, string s1, string s2);

##### Selects cover file by folder

[OperationContract]//(Action = "\*", ReplyAction = "\*")]

[FaultContract(typeof(MyApplicationFault))]

List<file2Info> SelectCoversByFolder(int folderId, string s1, string s2);

##### Select file info by fileId

[OperationContract]//(Action = "\*", ReplyAction = "\*")]

[FaultContract(typeof(MyApplicationFault))]

byte[] SelectFileById(int fileId, bool isVoucher, bool signed, int startFrom, string s1, string s2);

## Version update Methods

##### Retrieves version info by version string

[OperationContract]//(Action = "\*", ReplyAction = "\*")]

[FaultContract(typeof(MyApplicationFault))]

List<UpdateFileInfo> GetVersionInfo(string currentVersion, string s1, string s2);

##### Reads version of file by filename

[OperationContract]//(Action = "\*", ReplyAction = "\*")]

[FaultContract(typeof(MyApplicationFault))]

byte[] ReadVersionFile(string fileName, long from, string s1, string s2);

## Transfer file Methods

##### Function generates transfer file

[OperationContract]//(Action = "\*", ReplyAction = "\*")]

[FaultContract(typeof(MyApplicationFault))]

List<TransferFileInfo> GetTransferFile(int countryId, int beginNumber, int endNumber, string siteCode, string s1, string s2);

## General Methods

##### Retrieves any database table data

ArrayList RetrieveTableData(string fieldList, string tableName, string where, string s1, string s2);

##### Updates any database table data

int UpdateTableData(ArrayList table, string s1, string s2);

## TRS related Methods

##### Search for a voucher by Voucher number

[OperationContract]

[FaultContract(typeof(MyApplicationFault))]

VoucherInfo3 FindVoucherTRSByVoucherNumber(int countryId, int voucherId, string s1, string s2);

##### Search for a voucher by Sitecode in TRS database

[OperationContract]

[FaultContract(typeof(MyApplicationFault))]

VoucherInfo3 FindVoucherTRSBySiteCode(string siteCode, int location, string s1, string s2);

## PR related Methods

##### Search for a voucher by Sitecode in PR database

[OperationContract]

[FaultContract(typeof(MyApplicationFault))]

VoucherInfo3 FindVoucherPRBySiteCode(string siteCode, int location, string s1, string s2);

## Download Methods

##### Downloads Notadebito report by server url

[OperationContract]

[FaultContract(typeof(MyApplicationFault))]

byte[] DownloadReport(string rServerUrl, string s1, string s2);

##### Download vouchers by countryId, voucherId array

[OperationContract]

[FaultContract(typeof(MyApplicationFault))]

byte[] DownloadVouchers(int countryId, int[] voucherIds, string s1, string s2);

## Emailing Methods

##### Emails Notadebito by EmailInfo array

[OperationContract]

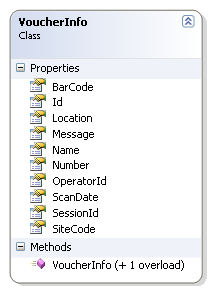
[FaultContract(typeof(MyApplicationFault))]

void EmailNotaDebito(EmailInfo[] emails, string s1, string s2);

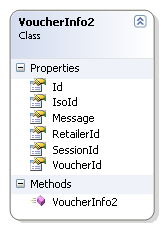
## Schema Objects

## VoucherInfo

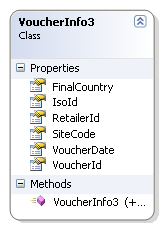
It provides data for processing voucher. It doesn’t care the image itself.



## VoucherInfo2



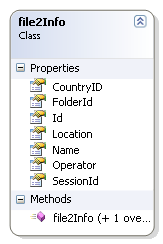
## VoucherInfo3



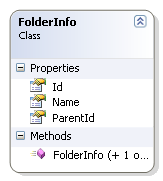
## fileInfo



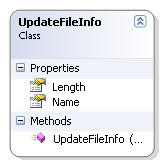
## file2Info



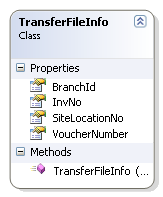
## FolderInfo



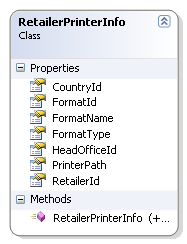
## UpdateFileInfo



## TransferFileInfo

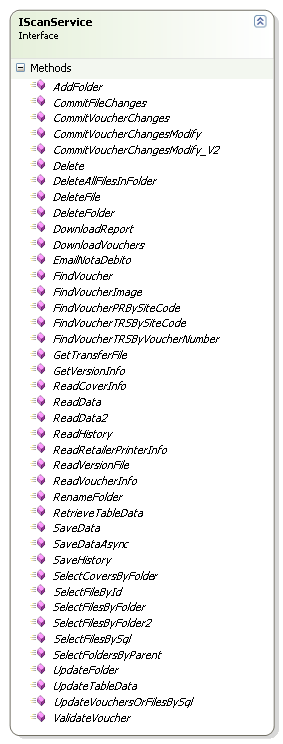


## RetailerPrinterInfo



## IScanService interface

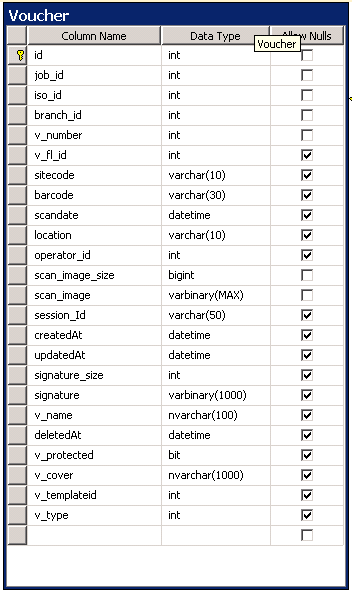
The interface WCF service uses to connect its clients.



## Table Description

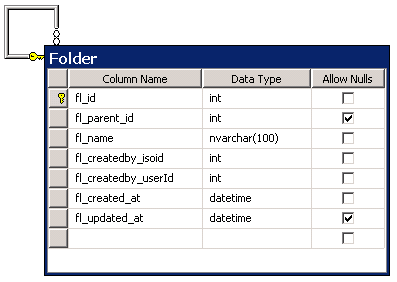
## Voucher table

It contains information about voucher images, scans and all Meta data taken during scan.



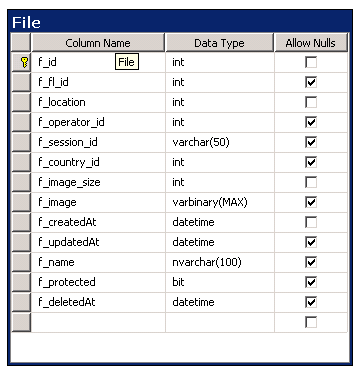
## Folder

It contains information about all folder structure of virtual image store of countries.



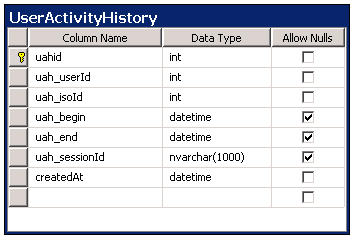
## File

It contains information about all additional files which come as attachments to the voucher.



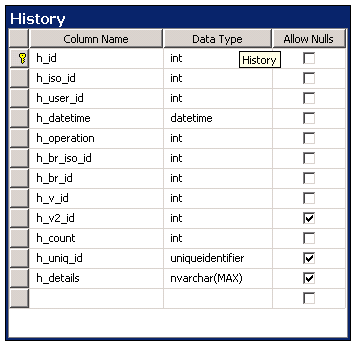
## User Activity History

Contains information about all operation performed by customers of MVC web site. Server records every showing of images or Meta data.



## History

Contains information about all operation performed from clients.



## Security

## Central Web server

It maintains security by requiring special string couple along the calls coming. Service computes the strings. In case of unauthorized access it throws the client security exception. Data coming with the request are also encrypted. Server decrypts data by using same algorithm and password the client use.

## File server

In maintain security by encrypting files before storing in files system. Files become unreadable. To unlock file server decrypt it by using same encrypting algorithm and password.

## Database

Database uses SQL authentication in connection string.

## Client

It maintains security by authentication clients against TRS Authentication service. In Login form system calls TRS LogInUser. TRS service returns string caring security account details. Operator can operated images for country they have been logged in.