# Design Document for User Story 5341

## As a user I want to access the TBI-CDS application and central database from the Internet.

## ASP .NET Pages

N/A

## ASP .NET User Controls

N/A

## ASP .NET Classes

### CData.cs

Code Type: C# Class

Uses Classes: CStatus

Uses PL/SQL Procedures: N/A

Uses MDWS Calls: N/A

Lines of Code: 72

* public CData(CData dataInit)
  + constructor for the base data class that takes a loaded CDATA
* public CData(CDataConnection conn,

string strClientIP,

long lUserID,

string strSessionID,

System.Web.SessionState.HttpSessionState SessionState,

bool bMDWSTransfer)

* + constructor for the base CData class
* public bool MDWSTransfer
  + is MDWS Transfer On?
* public System.Web.SessionState.HttpSessionState WebSession
  + web session passed in from caller
* public CDataConnection DBConn
  + database connection property
* public string ClientIP
  + get/set client ip
* public long UserID
  + user id of the user currently logged in
* public string SessionID
  + session id property
* public CStatus DBConnValid()
  + check for a valid connection

### CDataConnection.cs

Code Type: C# Class

Uses Classes: CDataUtils, StringBuilder, CDataParameterList, CStatus, OracleCommand, OracleParameter, byte[1], OracleConnection

Uses PL/SQL Procedures: PCK\_FX\_SEC.AuditTransaction, PCK\_SESSION.setSessionValue, PCK\_SESSION.clearAllSessionValues, PCK\_SESSION.deleteSessionValue, PCK\_SESSION.getSessionValue

Uses MDWS Calls: N/A

Lines of Code: 418

* private bool m\_bAudit;

public bool Audit

* + should we audit transactions?
* public CDataConnection()
  + constructor, clears member vars etc...
* public bool AuditTransaction(string strSPName,

CDataParameterList ParamList,

long lTxnStatusCode,

string strTxnStatusComment,

out long lStatusCode,

out string strStatus)

* + audits a transaction using the stored proc name and param list passed in
* public int ConnectionType
  + connection type property
* public string Status
  + status property
* public long StatusCode
  + status code property
* public bool SetSessionValue( long lAppUserID,

string strSessionSource,

string strKeySessionID,

string strKey,

string strKeyValue)

* + sets a session value in the database
* public bool ClearAllSessionValues( long lAppUserID,

string strSessionSource,

string strKeySessionID )

* + clears all session values cacheed in the database
* public bool ClearSessionValue( long lAppUserID,

string strSessionSource,

string strKeySessionID,

string strKey)

* + clears a session value cached in the database
* public bool GetSessionValue( long lAppUserID,

string strSessionSource,

string strKeySessionID,

string strKey,

out string strKeyValue)

* + gets a session value cached in the database
* ~CDataConnection()
  + CDataConnection desctructor, try to close and dispose
* public void Close()
  + close the connection
* public OracleConnection GetOracleConnection()
  + get the underlying oracle connection
* public CStatus ExecuteOracleSP(string strSPName,

CDataParameterList ParamList)

* + execute an oracle sp using the name and param list passed in. this version return status information in a CStatus object
* public bool ExecuteOracleSP(string strSPName,

CDataParameterList ParamList,

out long lStatusCode,

out string strStatus)

* + Executes an oracle stored proc using the SP name and parm list passed in. status code of zero means success
* public bool Connect(string strConnectionString, int nConnectionType, bool bAudit)
  + Connect to the database using the connection string and connection type passed in. if bAudit is true then all SP calls will be audited in the FX\_AUDIT table

### CDataConverter.cs

Code Type: C# Class

Uses Classes: JsonSerializer, CDataConverter, StringBuilder

Uses PL/SQL Procedures: N/A

Uses MDWS Calls: N/A

Lines of Code: 50

* public void WriteJsonDataRow(JsonWriter writer, DataRow dataRow)
  + write a Data Row as json
* public void WriteJsonDataTable(JsonWriter writer, DataTable dataTable)
  + write a data table as json
* public void WriteJsonDataSet(JsonWriter writer, DataSet dataSet)
  + write a dataset as json
* public string GetJsonDSString(DataSet dataSet)
  + get a dataset as a json string

### CDataParameter.cs

Code Type: C# Class

Uses Classes: N/A

Uses PL/SQL Procedures: N/A

Uses MDWS Calls: N/A

Lines of Code: 109

* public enum DataParameterType : int
  + possible types of parameters
* public CDataParameter()
  + constructor
* public override string ToString()
  + override for CDataParameter to string method
* public int ParameterType
  + parameter type
* public ParameterDirection Direction
  + parameter direction
* public string ParameterName
  + Parameter Name
* public long LongParameterValue
  + Parameter value - long
* public double DoubleParameterValue
  + parameter value - double
* public bool BoolParameterValue
  + Parameter value - bool
* public int IntParameterValue
  + Parameter value - int
* public string StringParameterValue
  + Parameter value - string
* public DateTime DateParameterValue
  + Parameter value - date
* public string CLOBParameterValue
  + Parameter value - CLOB
* public byte[] BLOBParameterValue
  + blob parameter value (byte array)

### CDataParameterList.cs

Code Type: C# Class

Uses Classes: CDataParameter, 1024]

Uses PL/SQL Procedures: N/A

Uses MDWS Calls: N/A

Lines of Code: 134

* public CDataParameterList()
  + constructor
* public long GetParamLongValue(string strParameterName)
  + returns a long paramater value by name
* public int GetParamIntValue(string strParameterName)
  + returns and int paramater by name
* public CDataParameterList(string strSessionID,

string strClientIP,

long lFXUserID)

* + construstor adds session interface paramaters to the list of params
* public bool AddInputParameter(string strName, DateTime dtValue)
  + add a date input parameter to the list
* public bool AddInputParameter(string strName, string strValue)
  + add a string input parameter to the list
* public bool AddInputParameterCLOB(string strName, string strValue)
  + add a clob(string) input parameter to the list
* public bool AddInputParameter(string strName, long lValue)
  + add a long input parameter to the list
* public bool AddInputParameter(string strName, bool bValue)
  + add a bool input parameter to the list
* public bool AddInputParameter(string strName, double dblValue)
  + add a double input parameter to the list
* public bool AddOutputParameter(string strName, DateTime dtValue)
  + add a date output parameter to the list
* public bool AddOutputParameter(string strName, string strValue)
  + add a string output parameter to the list
* public bool AddOutputParameter(string strName, long lValue)
  + add a long putput parameter to the list
* public bool AddOutputParameter(string strName, bool bValue)
  + add a bool output parameter to the list
* public bool AddOutputParameter(string strName, double dblValue)
  + add a double output parameter to the list
* public bool AddParameter(string strName, bool bValue, ParameterDirection pmDirection)
  + add a bool parameter to the list
* public bool AddParameter(string strName, double dblValue, ParameterDirection pmDirection)
  + add a double parameter to the list
* public bool AddParameter(string strName, DateTime dtValue, ParameterDirection pmDirection)
  + add a date parameter to the list
* public bool AddParameter(string strName, string strValue, ParameterDirection pmDirection)
  + add a string parameter to the list
* public bool AddCLOBParameter(string strName, string strValue, ParameterDirection pmDirection)
  + adds a clob(string) param to the list
* public bool AddParameter(string strName, long lValue, ParameterDirection pmDirection)
  + add a long parameter to the list
* public CDataParameter GetItemByName(string strParameterName)
  + get an item from the list by parameter name
* public CDataParameter GetItemByIndex(int nIndex)
  + get an item from the list by index
* public bool Add(CDataParameter parameter)
  + add a parameter to the list
* public bool AddBLOBParameter(string strName, byte[] byValue, ParameterDirection pmDirection)
  + add a blob paramateter to the list
* public bool AddInputParameterBLOB(string strName, byte[] byValue)
  + add a blob input parameter to the list
* public bool AddInputParameterBLOB(string strName,

System.Web.UI.HtmlControls.HtmlInputFile FileInput)

* + add a blob parameter to the list from an input control

### CDataSet.cs

Code Type: C# Class

Uses Classes: CStatus, CDataUtils, OleDbCommand, OracleParameter, DataSet, OracleDataAdapter

Uses PL/SQL Procedures: N/A

Uses MDWS Calls: N/A

Lines of Code: 192

* public string Status
  + get status
* public long StatusCode
  + get status code
* public CDataSet()
  + constructor
* ~CDataSet()
  + destructor
* public bool AppendData( CDataConnection conn,

DataSet dsFrom,

DataSet dsTo )

* + append all the columns and data from one ds (dsfrom) to another ds (dsto)
* public CStatus GetOracleDataSet(CDataConnection conn,

string strSPName,

CDataParameterList ParamList,

out DataSet ds)

* + gets a dataset using the connection, stored proc name and param list passed in. return a status object
* public DataSet GetOracleDataSet(CDataConnection conn,

string strSPName,

CDataParameterList ParamList,

out long lStatusCode,

out string strStatus)

* + gets a dataset using the connection, stored proc name and param list passed in. status code of zero means success

### CDataUtils.cs

Code Type: C# Class

Uses Classes: ListItem, Random, }, StringBuilder, DataTable

Uses PL/SQL Procedures: N/A

Uses MDWS Calls: N/A

Lines of Code: 777

* public CDataUtils()
  + constructor
* public string GetDateTimeAsString(DateTime dt)
  + used to retrieve a date/time in a string format for display
* public long GetDSLongValue(DataSet ds, string strField)
  + helper to get one value from a dataset as a long
* public string GetDSStringValue(DataSet ds, string strField)
  + helper to get one value from a dataset as a string value is sanitized to help prevent XSS
* public bool IsEmpty(DataSet ds)
  + helper to determine if a dataset is empty
* public DateTime GetDSDateTimeValue(DataSet ds, string strField)
  + helper to get a datetime value from a record
* public DateTime GetDSDateTimeValue(DataRow dr, string strField)
  + helper to get a datetime from a data row
* public string GetMonthAsText(int nMonth)
  + helper to get month abbreviation
* public string GetStringValueFromDS(DataSet ds, string strField)
  + helper to get 1 value from a dataset as a string value is sanitized to help prevent XSS
* public string GetDateValueAsStringFromDS(DataSet ds, string strField)
  + helper to get date value from a dataset as a string
* public long GetLongValueFromDS(DataSet ds, string strField)
  + helper to get 1 value from a dataset as a long
* public int GetIntValueFromDS(DataSet ds, string strField)
  + helper to get 1 value from a dataset as an integer
* public string GetCommaDelimitedValues(DataSet ds,

string strColumnName)

* + gets the values from a dataset and splits them with commas ','
* public string GetPipeDelimitedValues(DataSet ds,

string strColumnName)

* + gets the values from a dataset and splits them with pipes '|'
* public void LoadCheckListBox(CheckBoxList chklst, DataSet ds, StateBag sb)
  + load a .net checkboxlist from a dataset
* public void LoadComboBox(ListBox cboList, DataSet ds)
  + load a .net combo box
* public void LoadRadioButtonListBox(RadioButtonList rblList, DataSet ds)
  + load a .net radio button list box
* public void HandleCheckBoxListChange(CheckBoxList chklst, StateBag sb)
  + handle a .net checklistbox change
* public bool SetSelectedRadioListValue(RadioButtonList rblList, string strValue)
  + set .net radio button list
* public string GetRadioListBoxValue(RadioButtonList rblList, int nValuePieceIndex)
  + get selected radiobox value
* public bool SetSelectedCheckListboxValues(CheckBoxList chklst, string strValues)
  + sets the selected checkbox values
* public bool SetSelectedCheckListboxValues(CheckBoxList chklst, ListBox lst, string strValues)
  + sets the selected checkbox values
* public string GetUnSelectedCheckListBoxValues(CheckBoxList chklst, int nValuePieceIndex)
  + gets the selected checkbox values
* public string GetSelectedCheckListBoxValues(CheckBoxList chklst, int nValuePieceIndex)
  + gets the selected checkbox values
* public string GetSelectedCheckListBoxText(CheckBoxList chklst)
  + get selected checkboxlist text
* public string GetListBoxValues(ListBox lst, int nValuePieceIndex)
  + gets the selected checkbox values
* public string GetListBoxText(ListBox lst)
  + gets the selected checkbox values
* public string GetCheckYesNoText(CheckBox chk)
  + get Yes/No text from checkbox
* public string GetLongValueAsString(long lValue)
  + get a long value as a string
* public double GetCurrencyValue(double dblPct, double dblValue)
  + get currency value
* public double GetPercentValue(string strPercent)
  + get percent value
* public int GetIntValue(string strInt)
  + get int value
* public long GetLongValue(string strLong)
  + get long value
* public double GetDoubleValue(string strDouble)
  + get double value
* public double GetCurrencyValue(string strCurrency)
  + get currency value
* public long GetLongValue(bool bValue)
  + get a long value
* public string GetPercentAsString(double dblPercent)
  + get percent as string
* public string GenerateRandomNumber()
  + Get a random number, good for forcing the browser to refresh a page this is not cryptographically random
* public string GenerateRandomChars()
  + Get a random chars, good for forcing the browser to refresh a page this is not cryptographically random
* public void SetCheckBoxValue(CheckBox chk, long lValue)
  + set .net checkbox value
* public bool GetSSNElements(string strSSN,

out string strSSN1,

out string strSSN2,

out string strSSN3)

* + get ssn in pieces
* public DateTime GetDate(string strDateTime)
  + used to turn a mm/dd/yyyy string into a DateTime
* public string GetDateAsString(DateTime dt)
  + used to retrieve a date in a string format for display
* public string GetCurrentAgeAsString(DateTime dtDOB)
  + gets age as string from DOB passed in
* public void GetCurrentAge(DateTime dtDOB,

out int years,

out int months,

out int days)

* + gets years months data from dob passed in
* public void GetAge(DateTime dtDOB,

DateTime dtAsOf,

out int years,

out int months,

out int days)

* + gets years months days from dom and asof date
* public string NumberToRoman(int number, bool bUppercase)
  + Converts an integer value into Roman numerals
* public string GetLetterOutline(int number, bool bUpercase)
  + get letter for outlining
* public string GetJSArray(DataSet ds, string strFields)
  + gets a JS array from ds and fields
* public bool ValidateCompareDates(string strDate1,

string strDate2,

string strRelation,

out bool bIsValidDate1,

out bool bIsValidDate2,

out bool bComparison)

* + Validates two dates (string) and compares them ("before", "equal", "after")
* public string GetJSONString(DataSet ds)
  + getsjson string from ds
* public DataTable GetDataTable(DataSet ds)
  + gets a datatable from a dataset
* public bool OptionExists(DropDownList ctrl, string strOptionValue)
  + checks to see if option exists in ddl
* public bool OptionExists(CheckBoxList ctrl, string strOptionValue)
  + checks to see if option exists in checkboxlist
* public bool OptionExists(RadioButtonList ctrl, string strOptionValue)
  + checks to see if option exists in rbl
* public bool OptionExists(ListBox ctrl, string strOptionValue)
  + checks to see if option exists in listbox
* public bool OptionExists(ListControl ctrl, string strOptionValue)
  + checks to see if option exists in listcontrol

### CDataUtils2.cs

Code Type: C# Class

Uses Classes: Regex, CStatus, TripleDESCryptoServiceProvider, 3]

Uses PL/SQL Procedures: N/A

Uses MDWS Calls: N/A

Lines of Code: 351

* static public DateTime GetMDWSShortDate(string strDateTime)
  + get MDWS date without time
* public static string SanitizeCommand(string str)
  + sanitizes a command string
* public static string SanitizeEncodedHTML(string strEncoded)
  + Sanitize encoded HTML
* public static string Sanitize(string str)
  + sanitize a string
* public static string TruncateString(string strText, int nChars)
  + truncates the passed in string at the specifed length - 3 and adds an ellipsis
* static public string DelimitString(

string strText,

string strDelimiter,

int nChars)

* + breaks a string up and inserts a delimeter
* static public int GetRowCount(DataSet ds)
  + helper to get the number of rows in a dataset
* static public DateTime GetNullDate()
  + We use 01/01/0001 to mean a "null" date. this will return a null date
* static public bool IsNumeric(string strValue)
  + is this string a numeric value?
* static public bool IsDateNull(DateTime dt)
  + We use 01/01/0001 to mean a "null" date. this will check to see if the date is null
* static public k\_COMPARE CompareDates(DateTime dt1, DateTime dt2)
  + compare dates -1 = invalid 0 = equal 1 = dt1 > dt2 2 = dt2 > dt1
* static public k\_COMPARE CompareDateTimes(DateTime dt1, DateTime dt2)
  + compare datetimes -1 = invalid 0 = equal 1 = dt1 > dt2 2 = dt2 > dt1
* static public DateTime GetMDWSDate(string strDateTime)
  + gets a date time given a MDWS date formatted 20030428.000700
* static public DateTime GetDate(

string strDateTime,

int nHH,

int nMM,

int nSS)

* + gets a date given a formmated datetime string mm/dd/yyyy and hh mm ss
* static public DateTime GetDate(string strDateTime)
  + gets a date given a formmated datetime string mm/dd/yyyy
* static public long ToLong(string strValue)
  + convert a string to a long
* static public long ToLong(k\_ACTIVE activeValue)
  + converts a constant k\_active to a long
* static public string GetMilitaryDateAsString(DateTime dt)
  + gets mil date as string YYYYMMDD
* static public string GetDateAsString(DateTime dt)
  + used to retrieve a date in a string format for display
* static public string GetDateTimeAsString(DateTime dt)
  + used to retrieve a date/time in a string format for display
* static public bool IsEmpty(DataSet ds)
  + helper to determine if a dataset is empty
* static public CStatus GetDSDelimitedData(

DataSet ds,

string strField,

string strDelimeter,

out string strData)

* + US:838 gets a dataset as delimeted fields
* static public long GetDSLongValue(DataSet ds, string strField)
  + helper to get one value from a dataset as a string
* static public double GetDSDoubleValue(DataSet ds, string strField)
  + helper to get a double value from a DS
* static public double GetDSDoubleValue(DataRow dr, string strField)
  + helper to get a double value from a data row
* static public DateTime GetDSDateTimeValue(DataSet ds, string strField)
  + helper to get a datetime value from a record
* static public long GetDSLongValue(DataRow dr, string strField)
  + helper to get a value from a dataset row as a long
* static public string GetDSStringValue(DataRow dr, string strField)
  + helper to get one value from a dataset as a string
* static public DateTime GetDSDateTimeValue(DataRow dr, string strField)
  + helper to get a datetime from a data row
* static public string GetDSStringValue(DataSet ds, string strField)
  + helper to get one value from a dataset as a string
* static public k\_ACTIVE ToActiveID(string strValue)
  + converts a string to a contstant k\_ACTIVE
* static public k\_ACTIVE ToActiveID(int nValue)
  + converts an integer to a contstant k\_ACTIVE
* static public k\_ACTIVE ToActiveID(long lValue)
  + converts a long to a contstant k\_ACTIVE
* static public string enc(

string strClear,

string strEKey,

string strEInitVector)

* + triple des encrypt a string
* static public string dec(

string strEnc,

string strEKey,

string strEInitVector)

* + triple des decrypt a string

### CStatus.cs

Code Type: C# Class

Uses Classes: N/A

Uses PL/SQL Procedures: N/A

Uses MDWS Calls: N/A

Lines of Code: 26

* bool m\_bStatus;

public bool Status

* + bool status
* string m\_strStatusComment;

public string StatusComment

* + status comment
* k\_STATUS\_CODE m\_lStatusCode;

public k\_STATUS\_CODE StatusCode

* + status code
* public CStatus()
  + constructor
* public CStatus(bool bStatus, k\_STATUS\_CODE lStatusCode, string strStatusComment)
  + constructor initializes the members of the instance with the passed values

### CStringStatus.cs

Code Type: C# Class

Uses Classes: N/A

Uses PL/SQL Procedures: N/A

Uses MDWS Calls: N/A

Lines of Code: 18

* public string Value
  + gets/sets the value string
* public CStringStatus() : base()
  + constructor initializes the members of the instance with default values
* public CStringStatus(bool bStatus, k\_STATUS\_CODE lStatusCode, string strStatusComment, string strValue)

: base(bStatus, lStatusCode, strStatusComment)

* + constructor initializes the members of the instance with the passes values
* public CStringStatus(CStatus status, string strValue)

:base(status.Status, status.StatusCode, status.StatusComment)

* + constructor initializes the members of the instance with the passes values

### Enums.cs

Code Type: C# Class

Uses Classes: N/A

Uses PL/SQL Procedures: N/A

Uses MDWS Calls: N/A

Lines of Code: 64

* public enum k\_SOURCE\_TYPE\_ID : long
  + source type enum vista, vappct etc...
* public enum k\_SYSTEM : long
  + system id, used to identify transfer systems etc...
* public enum k\_DATA\_PARAMETER\_TYPE : int
  + enumeration of paramater types
* public enum k\_STATUS\_CODE : long
  + enumeration of status codes
* public enum k\_EDIT\_MODE : long
  + enumeration of edit modes
* public enum k\_EVENT : long
  + enumeration of events
* public enum k\_ACTIVE : long
  + enumeration of active/inactive
* public enum k\_ACTIVE\_ID : long
  + active enum, active, inactive all
* public enum k\_SEX : long
  + enumeration for sex
* public enum k\_COMPARE : long
  + enumeration for comparisons

## Oracle PL/SQL

### pck\_fx\_sec.bdy

Uses Tables: FX\_SEC, FX\_SESSION, FX\_SESSION\_VALUE, FX\_USER, FX\_USER\_ID, FX\_RESTRICTED\_NAME, UTL\_DECSTR, FX\_IPLOGIN\_ATTEMPTS, FX\_PAGE\_ACCESS, FX\_USER\_RIGHTS, FX\_AUDIT, UTL\_SEC.ENCRYPTDATA, FX\_SEC.UPDATESUATUSERRIGHTS, FX\_USER\_ID), STAT\_SEC\_QUESTIONS, FX\_SEC.GETSECURITYQUESTIONS, FX\_SEC\_QUESTIONS, FX\_SEC.GETUSERQUESTIONS, FX\_SEC.UPDATESECQUESTIONS, FX\_SEC.CHECKSECURITYQUESTIONS

Lines of Code: 1993

* procedure CreateSession(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number,

pi\_vWebSessionID in varchar2,

po\_vDBSessionID out varchar2,

po\_nStatusCode out number,

po\_vStatusComment out varchar2)

* + creates a database session for a user. used after a successful login to an external system such as MDWS to establish a session for the user in the database
* procedure ValidatePassword(pi\_vKey in varchar2,

pi\_nUserID in number,

pi\_vUserName in varchar2,

pi\_vOldPassword in varchar2,

pi\_vPassword in varchar2,

pi\_vCOldPassword in varchar2,

pi\_vCPassword in varchar2,

pi\_vCUserName in varchar2,

pi\_nResetPassword in number,

po\_nStatusCode out number,

po\_vStatusComment out varchar2)

* + validate the password against rules
* procedure DeleteAllSessionValues(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number,

po\_nStatusCode out number,

po\_vStatusComment out varchar2)

* + delete all session values
* procedure DeleteSessionValue(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number,

pi\_vKey in varchar2,

po\_nStatusCode out number,

po\_vStatusComment out varchar2)

* + delete 1 session value
* procedure GetSessionValue(pi\_vDBSessionID in varchar2,

pi\_vWebSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number,

pi\_vKey in varchar2,

po\_vKeyValue out varchar2,

po\_nStatusCode out number,

po\_vStatusComment out varchar2)

* + get a session value
* procedure SetSessionValue(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number,

pi\_vKey in varchar2,

pi\_vKeyValue in varchar2,

po\_nStatusCode out number,

po\_vStatusComment out varchar2)

* + set a session value
* procedure CertLogin(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number, not used but passed in to maintain consistant default param list

pi\_vCert in varchar2,

po\_nUserID out number,

po\_vDBSessionID out varchar2,

po\_nTimeout out number,

po\_nStatusCode out number,

po\_vStatusComment out varchar2)

* + cac cert login
* procedure ChangePassword(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number, not used but passed in to maintain consistant default param list

pi\_vKey in varchar2,

pi\_vUserName in varchar2,

pi\_vOldPassword in varchar2,

pi\_vPassword in varchar2,

pi\_vCert in varchar2,

pi\_vCOldPassword in varchar2,

pi\_vCPassword in varchar2,

pi\_vCUserName in varchar2,

po\_nUserID out number,

po\_vDBSessionID out varchar2,

po\_nTimeout out number,

po\_nStatusCode out number,

po\_vStatusComment out varchar2)

* + change password, this will also logg the user in
* procedure Sign(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number,

pi\_vUserName in varchar2,

pi\_vPassword in varchar2,

po\_vProviderID out varchar2,

po\_nUserType out number,

po\_nStatusCode out number,

po\_vStatusComment out varchar2)

* + used for signing notes etc.
* procedure Login(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number, not used but passed in to maintain consistant default param list

pi\_vUserName in varchar2,

pi\_vPassword in varchar2,

pi\_vCert in varchar2,

po\_nUserID out number,

po\_vDBSessionID out varchar2,

po\_nTimeout out number,

po\_nStatusCode out number,

po\_vStatusComment out varchar2)

* + login this is the version that requires a username and password after a successful login the persons CAC will be associated with this account
* procedure AuditPageAccess(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number,

pi\_vPageName in varchar2,

po\_nStatusCode out number,

po\_vStatusComment out varchar2)

* + audit page acceess
* procedure AuditTransaction(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number,

pi\_vSPName in varchar2,

pi\_clAuditXML in clob,

po\_nStatusCode out number,

po\_vStatusComment out varchar2)

* + audits a transaction by encrypting the clob passed in and storing the params and clob in the fx\_audit table
* procedure LogOff(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number,

po\_nStatusCode out number,

po\_vStatusComment out varchar2)

* + Log Off, clears session from the db
* procedure GetFXUserRS(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number,

pi\_vEncUID in varchar2,

po\_nStatusCode out number,

po\_vStatusComment out varchar2,

rs out RetRefCursor)

* + gets an fx\_user record given the encrypted uid
* procedure InsertFXUser(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number,

pi\_vKey in varchar2,

pi\_vProviderID in varchar2,

pi\_vUserName in varchar2,

pi\_vPassword in varchar2,

pi\_nAccountLocked in number,

pi\_nAccountInactive in number,

pi\_vCOldPassword in varchar2,

pi\_vCPassword in varchar2,

pi\_vCUserName in varchar2,

po\_nFXUserID out number,

po\_nStatusCode out number,

po\_vStatusComment out varchar2)

* + insert a record into the fx\_user table and update the fx\_user\_id in the suat\_user table
* procedure UpdateFXUser(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number,

pi\_nFXUserID in number,

pi\_vProviderID in varchar2,

pi\_vUserName in varchar2,

pi\_vPassword in varchar2,

pi\_nAccountLocked in number,

pi\_nAccountInactive in number,

po\_nStatusCode out number,

po\_vStatusComment out varchar2)

* + update an fx\_user record
* procedure UpdateFXUserRights(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number,

pi\_nFxUserID in number,

pi\_nUserType in number,

pi\_nUserRights in number,

pi\_nUserReadOnly in number,

po\_nStatusCode out number,

po\_vStatusComment out varchar2)

* + update fx\_user\_rights
* procedure UpdateFXUserPWD(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number,

pi\_vKey in varchar2,

pi\_nFXUserID in number,

pi\_vUserName in varchar2,

pi\_vPassword in varchar2,

pi\_nAccountLocked in number,

pi\_nAccountInactive in number,

pi\_vCPassword in varchar2,

pi\_vCUserName in varchar2,

po\_nStatusCode out number,

po\_vStatusComment out varchar2)

* + update an fx\_user record pwd
* procedure UpdateFXUserOptions(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number,

pi\_nFXUserID in number,

pi\_nAccountLocked in number,

pi\_nAccountInactive in number,

po\_nStatusCode out number,

po\_vStatusComment out varchar2)

* + update an fx\_user options
* procedure GetFXUsernamePasswordRS(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number,

pi\_vProviderID in varchar2,

po\_nStatusCode out number,

po\_vStatusComment out varchar2,

rs out RetRefCursor)

* + gets an Username and Password given the providerid
* procedure GetFXUserIdRS(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number,

pi\_vProviderID in varchar2,

po\_nStatusCode out number,

po\_vStatusComment out varchar2,

rs out RetRefCursor)

* + gets an FXUserID given the providerid
* procedure CheckFXUserRecRS(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number,

pi\_vProviderID in varchar2,

po\_nStatusCode out number,

po\_vStatusComment out varchar2,

rs out RetRefCursor)

* + gets a FXUserID given the providerid
* procedure GetSecurityQuestions(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number,

pi\_nQuestionGrp in number,

po\_nStatusCode out number,

po\_vStatusComment out varchar2,

rs out RetRefCursor)

* + gets security questions
* procedure GetUserQuestions(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number,

pi\_vUsername in varchar2,

po\_nStatusCode out number,

po\_vStatusComment out varchar2,

rs out RetRefCursor)

* + gets user questions
* procedure UpdateSecQuestions(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number,

pi\_nQuestionID\_1 in number,

pi\_vAnswer\_1 in varchar2,

pi\_nQuestionID\_2 in number,

pi\_vAnswer\_2 in varchar2,

pi\_nQuestionID\_3 in number,

pi\_vAnswer\_3 in varchar2,

po\_nStatusCode out number,

po\_vStatusComment out varchar2)

* + update security questions
* procedure CheckSecurityQuestions(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number,

pi\_nQuestionID\_1 in number,

pi\_vAnswer\_1 in varchar2,

pi\_nQuestionID\_2 in number,

pi\_vAnswer\_2 in varchar2,

pi\_nQuestionID\_3 in number,

pi\_vAnswer\_3 in varchar2,

po\_nStatusCode out number,

po\_vStatusComment out varchar2)

* + check security questions
* procedure ResetPassword(pi\_vSessionID in varchar2,

pi\_vSessionClientIP in varchar2,

pi\_nUserID in number,

pi\_vKey in varchar2,

pi\_nFXUserID in number,

pi\_vUserName in varchar2,

pi\_vPassword in varchar2,

pi\_vCPassword in varchar2,

pi\_vCUserName in varchar2,

po\_nStatusCode out number,

po\_vStatusComment out varchar2)

* + reset password