```
<%
function findChanges(oldEnrollment, newEnrollment)
    dim applicantId
        applicantId = oldEnrollment.ApplicantId

    dim cmd, changedInfo
    set cmd = Server.CreateObject("ADODB.Command")
    cmd.ActiveConnection = dsnLessTemps(oldEnrollment.SiteId)

    dim strWhatWasDone
        strWhatWasDone = "" &_
            "Applicant Information Updated by " &_
            "VMS User " & user_name & " (" & Request.ServerVariables("REMOTE_ADDR") & ")"

    cmd.CommandText = MakeAppointmentSQL(oldEnrollment, strWhatWasDone, applicantId)
    cmd.Execute()

    'Applicant
    InsertActivity oldEnrollment.LastnameFirst, newEnrollment.LastnameFirst, "LastnameFirst",
    applicantId, cmd
    InsertActivity oldEnrollment.Address, newEnrollment.Address, "Address", applicantId, cmd
    InsertActivity oldEnrollment.City, newEnrollment.City, "City", applicantId, cmd
    InsertActivity oldEnrollment.State, newEnrollment.State, "State", applicantId, cmd
    InsertActivity oldEnrollment.Zip, newEnrollment.Zip, "Zip", applicantId, cmd
    InsertActivity oldEnrollment.ApplicantStatus, newEnrollment.ApplicantStatus,
    "ApplicantStatus", applicantId, cmd
    InsertActivity oldEnrollment.Telephone, newEnrollment.Telephone, "Telephone", applicantId,
    cmd
    InsertActivity oldEnrollment.AltTelephone, newEnrollment.AltTelephone, "AltTelephone",
    applicantId, cmd
    InsertActivity oldEnrollment.ShortMemo, newEnrollment.ShortMemo, "ShortMemo", applicantId,
    cmd
    InsertActivity oldEnrollment.k, newEnrollment.k, "Skills", applicantId, cmd
    InsertActivity oldEnrollment.AppChangedDate, newEnrollment.AppChangedDate, "AppChangedDate"
    , applicantId, cmd
    InsertActivity oldEnrollment.EmailAddress, newEnrollment.EmailAddress, "EmailAddress",
    applicantId, cmd
    InsertActivity oldEnrollment.TaxJurisdiction, newEnrollment.TaxJurisdiction,
    "TaxJurisdiction", applicantId, cmd
    InsertActivity oldEnrollment.DateHired, newEnrollment.DateHired, "DateHired", applicantId,
    cmd
    InsertActivity oldEnrollment.Birthdate, newEnrollment.Birthdate, "Birthdate", applicantId,
    cmd
    InsertActivity oldEnrollment.MaritalStatus, newEnrollment.MaritalStatus, "MaritalStatus",
    applicantId, cmd
    InsertActivity oldEnrollment.EmpChangedDate, newEnrollment.EmpChangedDate, "EmpChangedDate"
    , applicantId, cmd
    InsertActivity oldEnrollment.StateExemptions, newEnrollment.StateExemptions,
    "StateExemptions", applicantId, cmd
    InsertActivity oldEnrollment.FedExemptions, newEnrollment.FedExemptions, "FedExemptions",
    applicantId, cmd
    InsertActivity oldEnrollment.EmpChangedBy, newEnrollment.EmpChangedBy, applicantId,
    "EmpChangedBy", cmd

    set cmd = nothing

end function

function CompareProperties(oldData, newData, propertyName, applicantId)

    if (oldData <> newData) then
        select case lcase(propertyName)
        case "skills"
```

```
            dim oldSkills
            oldSkills = split(oldData, ".")

            dim addedSkills
            addedSkills = newData

            dim skill
            for each skill in oldSkills
                if len(skill) > 0 then
                    addedSkills = replace(addedSkills, "." & skill, "")
                end if
            next

            dim newSkills
            newSkills = split(newData, ".")

            dim removedSkills
            removedSkills = oldData

            for each skill in newSkills
                if len(skill) > 0 then
                    removedSkills = replace(removedSkills, "." & skill, "")
                end if
            next

            CompareProperties = "" &_
                    "Changed [Skills] " &_
                    "added skills [" & addedSkills & "] " &_
                    "removed [" & removedSkills & "] " &_
                    "By ApplicantId: [" & applicantId & "]"

            set oldSkills = nothing
            set newSkills = nothing

        case else
            CompareProperties = "" &_
                "Changed [" & propertyName & "] " &_
                "from [" & oldData & "] " &_
                "to [" & newData & "] " &_
                "By ApplicantId: [" & applicantId & "]"

        end select
    end if

end function

function InsertActivity(oldData, newData, thisProperty, applicantId, cmd)
    dim changedInfo

    changedInfo = CompareProperties(oldData, newData, thisProperty, applicantId)
    if len(changedInfo) > 0 then
        cmd.CommandText = MakeAppointmentSQL(newEnrollment, changedInfo, applicantId)
        cmd.Execute()
    end if

end function

function MakeAppointmentSQL(objEnrollment, Comment, applicantId)
    const DispTypeCode = 3 'Disposistion status for 'TookPlace'
    Const ApptType = -1 'System Activity Type for 'Initial Interview'

        MakeAppointmentSQL =    "" &_
```

```
            "INSERT INTO Appointments " &_
                "(AppDate, ApplicantId, Comment, AssignedTo, ApptTypeCode, DispTypeCode,
                ContactId, Entered, EnteredBy, LocationId) " &_
                "VALUES (" &_
                    "#" & Date() & "#, " & _
                    insert_number(applicantId) & ", " &_
                    insert_string(Comment) & ", " &_
                    insert_string("{Anyone}") & ", " &_
                    insert_number(ApptType) & ", " &_
                    insert_number(DispTypeCode) & ", 0, " &_
                    "#" & Date() & "#, " & _
                    insert_string(tUser_id) & ", 1" &_
                ")"
end function

function cleanAddress(address)
    'for future development

end function

class cApplication

    private m_ApplicantId
    private m_ApplicationId
    Private m_ActiveConnection
    private m_SiteDSN
    private m_WhichCompany
    private m_WhereWeAre
    private m_MedicalProviders
    private m_LastnameFirst
    private m_PrevLastnameFirst
    private m_Lastname
    private m_Firstname
    private m_Address
    private m_AddressTwo
    private m_City
    private m_State
    private m_Zip
    private m_ApplicantStatus
    private m_Telephone
    private m_AltTelephone
    private m_ShortMemo
    private m_DateAvailable
    private m_Skills
    private m_AppChangedBy
    private m_AppChangedDate
    private m_EmailAddress
    private m_TaxJurisdiction
    private m_MaritalStatus
    private m_DOB
    private m_AliasNames
    private m_SSN
    private m_ApplicantName
    private m_Emailupdates
    private m_DesiredWageAmount
    private m_MinWageAmount
    private m_CurrentlyEmployed
    private m_Sex
    private m_Smoker
    private m_Citizen
    private m_AlienType
    private m_AlienNumber
    private m_WorkAuthProof
    private m_WorkTypeDesired
    private m_WorkAge
    private m_WorkValidLicense
    private m_WorkLicenseType
    private m_AutoInsurance
    private m_WorkRelocate
    private m_WorkConviction
    private m_WorkConvictionExplain
    private m_ECFullName
    private m_ECAddress
    private m_ECPhone
    private m_ECDoctor
    private m_ECDocPhone
    private m_W4a
    private m_W4b
    private m_W4c
    private m_W4d
    private m_W4e
    private m_W4f
    private m_W4g
    private m_W4h
    private m_W4more
    private m_MiddleInit
    private m_W4total
    private m_W4exempt
    private m_W4filing
    private m_W4namediffers
    private m_AvailableWhen
    private m_WorkCommuteHow
    private m_WorkCommuteDistance
    private m_WorkLicenseState
    private m_WorkLicenseExpire
    private m_WorkLicenseNumber
    private m_EduLevel
    private m_AdditionalInfo
    private m_SkillsSet
    private m_EmployerNameHistOne
    private m_JobHistAddOne
    private m_JobHistPhoneOne
    private m_JobHistPayOne
    private m_JobHistCityStateZipOne
    private m_JobHistSupervisorOne
    private m_JobDutiesOne
    private m_JobHistFromDateOne
    private m_JobHistToDateOne
    private m_JobReasonOne
    private m_EmployerNameHistTwo
    private m_JobHistAddTwo
    private m_JobHistPhoneTwo
    private m_JobHistPayTwo
    private m_JobHistCityStateZipTwo
    private m_JobHistSupervisorTwo
    private m_JobDutiesTwo
    private m_JobHistFromDateTwo
    private m_JobHistToDateTwo
    private m_JobReasonTwo
    private m_EmployerNameHistThree
    private m_JobHistAddThree
    private m_JobHistPhoneThree
    private m_JobHistPayThree
    private m_JobHistCityStateZipThree
    private m_JobHistSupervisorThree
    private m_JobDutiesThree
    private m_JobHistFromDateThree
    private m_JobHistToDateThree
    private m_JobReasonThree
    private m_PandPAgree
    private m_UnEmpAgree
    private m_ApplicantAgree
    private m_Username
    private m_UserId
    private m_Enrollments
    private m_DateLastEnrolled
    private m_DateLastModified
    private m_DateCreated


    sub Class_Initialize()
        set m_MedicalProviders = Server.CreateObject("Scripting.Dictionary")
        set m_Enrollments      = Server.CreateObject("Scripting.Dictionary")
    end sub
    sub Class_Terminate()
        set m_MedicalProviders = nothing
        set m_Enrollments      = nothing
    end sub

    public property get ApplicantId
        ApplicantId = m_ApplicantId
    end property
    public property let ApplicantId(p_ApplicantId)
        m_ApplicantId = p_ApplicantId
    end property

    public property get ApplicationId
        ApplicationId = m_ApplicationId
    end property
    public property let ApplicationId(p_ApplicationId)
        m_ApplicationId = p_ApplicationId
    end property

    public property get ActiveConnection
        ActiveConnection = m_ActiveConnection
    end property
    public property let ActiveConnection(p_ActiveConnection)
        m_ActiveConnection = p_ActiveConnection
    end property

    public property get SiteDSN()
        SiteDSN = m_SiteDSN
    end property
    public property let SiteDSN(p_SiteDSN)
        m_SiteDSN = p_SiteDSN
    end property
    public property get WhichCompany()
        WhichCompany = m_WhichCompany
    end property
    public property let WhichCompany(p_WhichCompany)
        m_WhichCompany = p_WhichCompany
    end property

    public property get WhereWeAre
        WhereWeAre = m_WhereWeAre
    end property
    public property let WhereWeAre(p_WhereWeAre)
        m_WhereWeAre = p_WhereWeAre
    end property
    public property get MedicalProviders()
        set MedicalProviders = m_MedicalProviders
    end property
    public property set MedicalProviders(p_MedicalProviders)
        set m_MedicalProviders = p_MedicalProviders
    end property

    public property get LastnameFirst()
        LastnameFirst = me.Lastname & ", " & me.Firstname
    end property

    public property get PrevLastnameFirst()
        PrevLastnameFirst = m_PrevLastnameFirst
    end property
    public property let PrevLastnameFirst(p_PrevLastnameFirst)
        m_PrevLastnameFirst = p_PrevLastnameFirst
    end property

    public property get ApplicantName()
        ApplicantName = me.Firstname & " " & me.Lastname
    end property

    public property get Lastname()
        Lastname = m_Lastname
    end property
    public property let Lastname(p_Lastname)
        m_Lastname = PCase(StripCharacters(p_Lastname))
    end property

    public property get Firstname()
        Firstname = m_Firstname
    end property
    public property let Firstname(p_Firstname)
        m_Firstname = PCase(StripCharacters(p_Firstname))
    end property

    public property get CityStateZip()
        CityStateZip = me.City & ", " & me.State & " " & me.Zip
    end property

    public property get LongAddress()
        if Len(me.AddressTwo) > 0 Then
            LongAddress = me.Address & ", " & me.AddressTwo & ", " & me.CityStateZip
        else
            LongAddress = me.Address & ", " & me.CityStateZip
        end if
    end property

    public property get Address()
        Address = m_Address
    end property
    public property let Address(p_Address)
        m_Address = StripCharacters(p_Address)
    end property

    public property get AddressTwo()
        AddressTwo = m_AddressTwo
    end property
    public property let AddressTwo(p_AddressTwo)
        m_AddressTwo = StripCharacters(p_AddressTwo)
    end property
    public property get City()
        City = m_City
    end property
    public property let City(p_City)
        m_City = StripCharacters(p_City)
    end property

    public property get State()
        State = m_State
    end property
    public property let State(p_State)
        m_State = StripCharacters(p_State)
    end property

    public property get Zip()
        Zip = m_Zip
    end property
    public property let Zip(p_Zip)
        m_Zip = only_zipcode(p_Zip)
    end property

    public property get ApplicantStatus()
        ApplicantStatus = m_ApplicantStatus
    end property
    public property let ApplicantStatus(p_ApplicantStatus)
        m_ApplicantStatus = p_ApplicantStatus
    end property

    public property get Telephone()
        Telephone = m_Telephone
    end property
    public property let Telephone(p_Telephone)
        m_Telephone = FormatPhoneNumber(p_Telephone)
    end property

    public property get AltTelephone()
        AltTelephone = m_AltTelephone
    end property
    public property let AltTelephone(p_AltTelephone)
        m_AltTelephone = FormatPhoneNumber(p_AltTelephone)
    end property

    public property get ShortMemo()
        ShortMemo = m_ShortMemo
    end property
    public property let ShortMemo(p_ShortMemo)
        m_ShortMemo = p_ShortMemo
    end property

    public property get DateAvailable()
        DateAvailable = m_DateAvailable
    end property
    public property let DateAvailable(p_DateAvailable)
        m_DateAvailable = p_DateAvailable
    end property

    public property get Skills()
        Skills = m_Skills
    end property
    public property let Skills(p_Skills)
        m_Skills = p_Skills
    end property
    public property get AppChangedBy()
        AppChangedBy = m_AppChangedBy
    end property
    public property let AppChangedBy(p_AppChangedBy)
        m_AppChangedBy = p_AppChangedBy
    end property

    public property get AppChangedDate()
        AppChangedDate = m_AppChangedDate
    end property
    public property let AppChangedDate(p_AppChangedDate)
        m_AppChangedDate = p_AppChangedDate
    end property

    public property get EmailAddress()
        EmailAddress = m_EmailAddress
    end property
    public property let EmailAddress(p_EmailAddress)
        m_EmailAddress = p_EmailAddress
    end property

    public property get TaxJurisdiction()
        TaxJurisdiction = m_TaxJurisdiction
    end property
    public property let TaxJurisdiction(p_TaxJurisdiction)
        m_TaxJurisdiction = p_TaxJurisdiction
    end property

    public property get MaritalStatus()
        MaritalStatus = m_MaritalStatus
    end property
    public property let MaritalStatus(p_MaritalStatus)
        m_MaritalStatus = StripCharacters(p_MaritalStatus)
    end property

    public property get DOB()
        DOB = m_DOB
    end property
    public property let DOB(p_DOB)
        if isDate(p_DOB) then
            m_DOB = FormatDateTime(p_DOB, 2)
        end if
    end property

    public property get AliasNames()
        AliasNames = m_AliasNames
    end property
    public property let AliasNames(p_AliasNames)
        m_AliasNames = p_AliasNames
    end property

    public property get SSN()
        SSN = m_SSN
    end property
    public property let SSN(p_SSN)

        if len(p_SSN) > 0 then
            dim ssnRE
            set ssnRE = New RegExp
            with ssnRE
                .Pattern = "[()-.<>'$\s]"
                .Global = True
            end with
            m_SSN = ssnRE.Replace(p_SSN, "")
        end if

    end property

    public property get EmailUpdates()
        EmailUpdates = m_EmailUpdates
    end property
    public property let EmailUpdates(p_EmailUpdates)
        m_EmailUpdates = StripCharacters(p_EmailUpdates)
    end property

    public property get DesiredWageAmount()
        DesiredWageAmount = m_DesiredWageAmount
    end property
    public property let DesiredWageAmount(p_DesiredWageAmount)
        m_DesiredWageAmount = StripCharacters(p_DesiredWageAmount)
    end property

    public property get MinWageAmount()
        MinWageAmount = m_MinWageAmount
    end property
    public property let MinWageAmount(p_MinWageAmount)
        m_MinWageAmount = StripCharacters(p_MinWageAmount)
    end property

    public property get CurrentlyEmployed()
        CurrentlyEmployed = m_CurrentlyEmployed
    end property
    public property let CurrentlyEmployed(p_CurrentlyEmployed)
        m_CurrentlyEmployed = StripCharacters(p_CurrentlyEmployed)
    end property

    public property get Sex()
        Sex = m_Sex
    end property
    public property let Sex(p_Sex)
        m_Sex = StripCharacters(p_Sex)
    end property

    public property get Smoker()
        Smoker = m_Smoker
    end property
    public property let Smoker(p_Smoker)
        m_Smoker = p_Smoker
    end property

    public property get Citizen()
        Citizen = m_Citizen
    end property
    public property let Citizen(p_Citizen)
        m_Citizen = p_Citizen
    end property

    public property get AlienType()
        AlienType = m_AlienType
    end property
    public property let AlienType(p_AlienType)
        m_AlienType = p_AlienType
    end property

    public property get AlienNumber()
        AlienNumber = m_AlienNumber
    end property
    public property let AlienNumber(p_AlienNumber)
        m_AlienNumber = p_AlienNumber
    end property

    public property get WorkAuthProof()
        WorkAuthProof = m_WorkAuthProof
    end property
    public property let WorkAuthProof(p_WorkAuthProof)
        m_WorkAuthProof = p_WorkAuthProof
    end property

    public property get WorkTypeDesired()
        WorkTypeDesired = m_WorkTypeDesired
    end property
    public property let WorkTypeDesired(p_WorkTypeDesired)
        m_WorkTypeDesired = StripCharacters(p_WorkTypeDesired)
    end property

    public property get WorkAge()
        WorkAge = m_WorkAge
    end property
    public property let WorkAge(p_WorkAge)
        m_WorkAge = p_WorkAge
    end property

    public property get WorkValidLicense()
        WorkValidLicense = m_WorkValidLicense
    end property
    public property let WorkValidLicense(p_WorkValidLicense)
        m_WorkValidLicense = p_WorkValidLicense
    end property

    public property get WorkLicenseType()
        WorkLicenseType = m_WorkLicenseType
    end property
    public property let WorkLicenseType(p_WorkLicenseType)
        m_WorkLicenseType = p_WorkLicenseType
    end property

    public property get AutoInsurance()
        AutoInsurance = m_AutoInsurance
    end property
    public property let AutoInsurance(p_AutoInsurance)
        m_AutoInsurance = p_AutoInsurance
    end property

    public property get WorkRelocate()
        WorkRelocate = m_WorkRelocate
    end property
    public property let WorkRelocate(p_WorkRelocate)
        m_WorkRelocate = StripCharacters(p_WorkRelocate)
    end property

    public property get WorkConviction()
        WorkConviction = m_WorkConviction
    end property
    public property let WorkConviction(p_WorkConviction)
        m_WorkConviction = StripCharacters(p_WorkConviction)
    end property

    public property get WorkConvictionExplain()
        WorkConvictionExplain = m_WorkConvictionExplain
    end property
    public property let WorkConvictionExplain(p_WorkConvictionExplain)
        m_WorkConvictionExplain = p_WorkConvictionExplain
    end property

    public property get ECFullName()
        ECFullName = m_ECFullName
    end property
    public property let ECFullName(p_ECFullName)
        m_ECFullName = p_ECFullName
    end property

    public property get ECAddress()
        ECAddress = m_ECAddress
    end property
    public property let ECAddress(p_ECAddress)
        m_ECAddress = p_ECAddress
    end property

    public property get ECPhone()
        ECPhone = m_ECPhone
    end property
    public property let ECPhone(p_ECPhone)

        if not isnull(p_ECPhone) then
            if len(p_ECPhone) > 0 then
                m_ECPhone = formatPhone(p_ECPhone)
            end if
        end if

    end property

    public property get ECDoctor()
        ECDoctor = m_ECDoctor
    end property
    public property let ECDoctor(p_ECDoctor)
        m_ECDoctor = p_ECDoctor
    end property

    public property get ECDocPhone()
        ECDocPhone = m_ECDocPhone
    end property
    public property let ECDocPhone(p_ECDocPhone)
        m_ECDocPhone = p_ECDocPhone
    end property

    public property get W4a()
        W4a = m_W4a
    end property
    public property let W4a(p_W4a)

        if isnull(p_W4a) then
            W4a = 0
        else
            m_W4a = p_W4a
        end if

    end property

    public property get W4b()
        W4b = m_W4b
    end property
    public property let W4b(p_W4b)
```

```
        if isnull(p_W4b) then
            W4b = 0
        else
            m_W4b = p_W4b
        end if

    end property

    public property get W4c()
        W4c = m_W4c
    end property
    public property let W4c(p_W4c)
        if isnull(p_W4c) then
            W4c = 0
        else
            m_W4c = p_W4c
        end if
    end property

    public property get W4d()
        W4d = m_W4d
    end property
    public property let W4d(p_W4d)
        if isnull(p_W4d) then
            W4d = 0
        else
            m_W4d = p_W4d
        end if
    end property

    public property get W4e()
        W4e = m_W4e
    end property
    public property let W4e(p_W4e)
        if isnull(p_W4e) then
            W4e = 0
        else
            m_W4e = p_W4e
        end if
    end property

    public property get W4f()
        W4f = m_W4f
    end property
    public property let W4f(p_W4f)
        if isnull(p_W4f) then
            W4f = 0
        else
            m_W4f = p_W4f
        end if
    end property

    public property get W4g()
        W4g = m_W4g
    end property
    public property let W4g(p_W4g)
        m_W4g = p_W4g
    end property

    public property get W4h()
        W4h= m_W4h
    end property
    public property let W4h(p_W4h)
        if isnull(p_W4h) then
            W4h = 0
        else
            m_W4h = p_W4h
        end if
    end property

    public property get W4more()
        W4more = m_W4more
    end property
    public property let W4more(p_W4more)
        m_W4more = p_W4more
    end property

    public property get MiddleInit()
        MiddleInit = m_MiddleInit
    end property
    public property let MiddleInit(p_MiddleInit)
        m_MiddleInit = p_MiddleInit
    end property

    public property get W4Total()
        W4Total = cint(m_W4Total)
    end property
    public property let W4Total(p_W4Total)

        if not isnull(p_W4Total) then
            m_W4Total = p_W4Total
        else
            m_W4Total = 0
        end if

    end property

    public property get W4Exempt()
        W4Exempt = cint(m_W4Exempt)
    end property
    public property let W4Exempt(p_W4Exempt)

        if not isnull(p_W4Exempt) then
            m_W4Exempt = p_W4Exempt
        else
            m_W4Exempt = 0
        end if

    end property

    public property get W4Filing()
        W4Filing = cint(m_W4Filing)
    end property
    public property let W4Filing(p_W4Filing)

        if not isnull(p_W4Filing) then
            m_W4Filing = p_W4Filing
        else
            m_W4Filing = 0
        end if

    end property

    public property get W4NameDiffers()
        W4NameDiffers = m_W4NameDiffers
    end property
    public property let W4NameDiffers(p_W4NameDiffers)
        m_W4NameDiffers = p_W4NameDiffers
    end property

    public property get AvailableWhen()
        AvailableWhen = m_AvailableWhen
    end property
    public property let AvailableWhen(p_AvailableWhen)
        m_AvailableWhen = p_AvailableWhen
    end property

    public property get WorkCommuteHow()
        WorkCommuteHow = m_WorkCommuteHow
    end property
    public property let WorkCommuteHow(p_WorkCommuteHow)
        m_WorkCommuteHow = p_WorkCommuteHow
    end property

    public property get WorkCommuteDistance()
        WorkCommuteDistance = m_WorkCommuteDistance
    end property
    public property let WorkCommuteDistance(p_WorkCommuteDistance)
        m_WorkCommuteDistance = p_WorkCommuteDistance
    end property

    public property get WorkLicenseState()
        WorkLicenseState = m_WorkLicenseState
    end property
    public property let WorkLicenseState(p_WorkLicenseState)
        m_WorkLicenseState = p_WorkLicenseState
    end property

    public property get WorkLicenseExpire()
        WorkLicenseExpire = m_WorkLicenseExpire
    end property
    public property let WorkLicenseExpire(p_WorkLicenseExpire)
        m_WorkLicenseExpire = p_WorkLicenseExpire
    end property

    public property get WorkLicenseNumber()
        WorkLicenseNumber = m_WorkLicenseNumber
    end property
    public property let WorkLicenseNumber(p_WorkLicenseNumber)
        m_WorkLicenseNumber = p_WorkLicenseNumber
    end property

    public property get EduLevel()
        EduLevel = m_EduLevel
    end property
    public property let EduLevel(p_EduLevel)
        m_EduLevel = StripCharacters(p_EduLevel)
    end property

    public property get AdditionalInfo()
        AdditionalInfo = m_AdditionalInfo
    end property
    public property let AdditionalInfo(p_AdditionalInfo)
        m_AdditionalInfo = StripCharacters(p_AdditionalInfo)
    end property

    public property get SkillsSet()
        SkillsSet = m_SkillsSet
    end property
    public property let SkillsSet(p_SkillsSet)
        m_SkillsSet = p_SkillsSet
    end property

    public property get EmployerNameHistOne()
        EmployerNameHistOne = m_EmployerNameHistOne
    end property
    public property let EmployerNameHistOne(p_EmployerNameHistOne)
        m_EmployerNameHistOne = PCase(StripCharacters(p_EmployerNameHistOne))
    end property

    public property get JobHistAddOne()
        JobHistAddOne = m_JobHistAddOne
    end property
    public property let JobHistAddOne(p_JobHistAddOne)
        m_JobHistAddOne = StripCharacters(p_JobHistAddOne)
    end property

    public property get JobHistPhoneOne()
        JobHistPhoneOne = m_JobHistPhoneOne
    end property
    public property let JobHistPhoneOne(p_JobHistPhoneOne)
        m_JobHistPhoneOne = p_JobHistPhoneOne
    end property

    public property get JobHistPayOne()
        JobHistPayOne = m_JobHistPayOne
    end property
    public property let JobHistPayOne(p_JobHistPayOne)
        m_JobHistPayOne = StripCharacters(p_JobHistPayOne)
    end property

    public property get JobHistCityStateZipOne()
        JobHistCityStateZipOne = m_JobHistCityStateZipOne
    end property
    public property let JobHistCityStateZipOne(p_JobHistCityStateZipOne)
        m_JobHistCityStateZipOne = StripCharacters(p_JobHistCityStateZipOne)
    end property

    public property get JobHistSupervisorOne()
        JobHistSupervisorOne = m_JobHistSupervisorOne
    end property
    public property let JobHistSupervisorOne(p_JobHistSupervisorOne)
        m_JobHistSupervisorOne = StripCharacters(p_JobHistSupervisorOne)
    end property

    public property get JobDutiesOne()
        JobDutiesOne = m_JobDutiesOne
    end property
    public property let JobDutiesOne(p_JobDutiesOne)
        m_JobDutiesOne = p_JobDutiesOne
    end property

    public property get JobHistFromDateOne()
        JobHistFromDateOne = m_JobHistFromDateOne
    end property
    public property let JobHistFromDateOne(p_JobHistFromDateOne)
        m_JobHistFromDateOne = StripCharacters(p_JobHistFromDateOne)
    end property

    public property get JobHistToDateOne()
        JobHistToDateOne = m_JobHistToDateOne
    end property
    public property let JobHistToDateOne(p_JobHistToDateOne)
        m_JobHistToDateOne = StripCharacters(p_JobHistToDateOne)
    end property

    public property get JobReasonOne()
        JobReasonOne = m_JobReasonOne
    end property
    public property let JobReasonOne(p_JobReasonOne)
        m_JobReasonOne = p_JobReasonOne
    end property

    public property get EmployerNameHistTwo()
        EmployerNameHistTwo = m_EmployerNameHistTwo
    end property
    public property let EmployerNameHistTwo(p_EmployerNameHistTwo)
        m_EmployerNameHistTwo = PCase(StripCharacters(p_EmployerNameHistTwo))
    end property

    public property get JobHistAddTwo()
        JobHistAddTwo = m_JobHistAddTwo
    end property
    public property let JobHistAddTwo(p_JobHistAddTwo)
        m_JobHistAddTwo = StripCharacters(p_JobHistAddTwo)
    end property

    public property get JobHistPhoneTwo()
        JobHistPhoneTwo = m_JobHistPhoneTwo
    end property
    public property let JobHistPhoneTwo(p_JobHistPhoneTwo)
        m_JobHistPhoneTwo = StripCharacters(p_JobHistPhoneTwo)
    end property

    public property get JobHistPayTwo()
        JobHistPayTwo = m_JobHistPayTwo
    end property
    public property let JobHistPayTwo(p_JobHistPayTwo)
        m_JobHistPayTwo = StripCharacters(p_JobHistPayTwo)
    end property

    public property get JobHistCityStateZipTwo()
        JobHistCityStateZipTwo = m_JobHistCityStateZipTwo
    end property
    public property let JobHistCityStateZipTwo(p_JobHistCityStateZipTwo)
        m_JobHistCityStateZipTwo = StripCharacters(p_JobHistCityStateZipTwo)
    end property

    public property get JobHistSupervisorTwo()
        JobHistSupervisorTwo = m_JobHistSupervisorTwo
    end property
    public property let JobHistSupervisorTwo(p_JobHistSupervisorTwo)
        m_JobHistSupervisorTwo = StripCharacters(p_JobHistSupervisorTwo)
    end property

    public property get JobDutiesTwo()
        JobDutiesTwo = m_JobDutiesTwo
    end property
    public property let JobDutiesTwo(p_JobDutiesTwo)
        m_JobDutiesTwo = p_JobDutiesTwo
    end property

    public property get JobHistFromDateTwo()
        JobHistFromDateTwo = m_JobHistFromDateTwo
    end property
    public property let JobHistFromDateTwo(p_JobHistFromDateTwo)
        m_JobHistFromDateTwo = StripCharacters(p_JobHistFromDateTwo)
    end property

    public property get JobHistToDateTwo()
        JobHistToDateTwo = m_JobHistToDateTwo
    end property
    public property let JobHistToDateTwo(p_JobHistToDateTwo)
        m_JobHistToDateTwo = StripCharacters(p_JobHistToDateTwo)
    end property

    public property get JobReasonTwo()
        JobReasonTwo = m_JobReasonTwo
    end property
    public property let JobReasonTwo(p_JobReasonTwo)
        m_JobReasonTwo = p_JobReasonTwo
    end property

    public property get EmployerNameHistThree()
        EmployerNameHistThree = m_EmployerNameHistThree
    end property
    public property let EmployerNameHistThree(p_EmployerNameHistThree)
        m_EmployerNameHistThree = PCase(StripCharacters(p_EmployerNameHistThree))
    end property

    public property get JobHistAddThree()
        JobHistAddThree = m_JobHistAddThree
    end property
    public property let JobHistAddThree(p_JobHistAddThree)
        m_JobHistAddThree = StripCharacters(p_JobHistAddThree)
    end property

    public property get JobHistPhoneThree()
        JobHistPhoneThree = m_JobHistPhoneThree
    end property
    public property let JobHistPhoneThree(p_JobHistPhoneThree)
        m_JobHistPhoneThree = StripCharacters(p_JobHistPhoneThree)
    end property

    public property get JobHistPayThree()
        JobHistPayThree = m_JobHistPayThree
    end property
    public property let JobHistPayThree(p_JobHistPayThree)
        m_JobHistPayThree = StripCharacters(p_JobHistPayThree)
    end property

    public property get JobHistCityStateZipThree()
        JobHistCityStateZipThree = m_JobHistCityStateZipThree
    end property
    public property let JobHistCityStateZipThree(p_JobHistCityStateZipThree)
        m_JobHistCityStateZipThree = StripCharacters(p_JobHistCityStateZipThree)
    end property

    public property get JobHistSupervisorThree()
        JobHistSupervisorThree = m_JobHistSupervisorThree
    end property
    public property let JobHistSupervisorThree(p_JobHistSupervisorThree)
        m_JobHistSupervisorThree = StripCharacters(p_JobHistSupervisorThree)
    end property

    public property get JobDutiesThree()
        JobDutiesThree = m_JobDutiesThree
    end property
    public property let JobDutiesThree(p_JobDutiesThree)
        m_JobDutiesThree = p_JobDutiesThree
    end property

    public property get JobHistFromDateThree()
        JobHistFromDateThree = m_JobHistFromDateThree
    end property
    public property let JobHistFromDateThree(p_JobHistFromDateThree)
        m_JobHistFromDateThree = StripCharacters(p_JobHistFromDateThree)
    end property

    public property get JobHistToDateThree()
        JobHistToDateThree = m_JobHistToDateThree
    end property
    public property let JobHistToDateThree(p_JobHistToDateThree)
        m_JobHistToDateThree = StripCharacters(p_JobHistToDateThree)
    end property

    public property get JobReasonThree()
        JobReasonThree = m_JobReasonThree
    end property
    public property let JobReasonThree(p_JobReasonThree)
        m_JobReasonThree = p_JobReasonThree
    end property

    public property get PAndPAgree()
        PAndPAgree = m_PAndPAgree
    end property
    public property let PAndPAgree(p_PAndPAgree)
        m_PAndPAgree = p_PAndPAgree
    end property

    public property get UnEmpAgree()
        UnEmpAgree = m_UnEmpAgree
    end property
    public property let UnEmpAgree(p_UnEmpAgree)
        m_UnEmpAgree = p_UnEmpAgree
    end property

    public property get ApplicantAgree()
        ApplicantAgree = m_ApplicantAgree
    end property
    public property let ApplicantAgree(p_ApplicantAgree)
        m_ApplicantAgree = p_ApplicantAgree
    end property

    public property get UserName()
        UserName = m_UserName
    end property
    public property let UserName(p_UserName)
        m_UserName = p_UserName
    end property

    public property get UserId
        UserId = m_UserId
    end property
    public property let UserId(p_UserId)
        m_UserId = p_UserId
    end property

    public property get Enrollments()
        set Enrollments = m_Enrollments
    end property
    public property let Enrollments(p_Enrollments)
        set m_Enrollments = p_Enrollments
    end property

    public property get DateLastEnrolled
        DateLastEnrolled = m_DateLastEnrolled
    end property
    public property let DateLastEnrolled(p_DateLastEnrolled)
        m_DateLastEnrolled = p_DateLastEnrolled
    end property

    public property get DateLastModified
        DateLastModified = m_DateLastModified
    end property
    public property let DateLastModified(p_DateLastModified)
        DateLastModified = m_ActiveConnection
    end property

    public property get DateCreated
        DateCreated = m_DateCreated
    end property
    public property let DateCreated(p_DateCreated)
        m_DateCreated = p_DateCreated
    end property


'#############  Public Functions ##############

    public function GetSiteAddress()

        GetSiteAddress = Left(Request.ServerVariables("REMOTE_ADDR"), 9)

    end function

    public function WhereAreWe()

    end function

    public function LoadApplication()

        dim sqlMedicalProviders
        sqlMedicalProviders = "" &_
            "SELECT tbl_siteips.site, tbl_siteips.locationId, med_providers.name, " &_
                "med_providers.address, med_providers.phone, med_providers.fax, " &_
                "med_providers.cityandstate, med_providers.openhours, med_providers.id " &_
            "FROM tbl_siteips INNER JOIN med_providers ON tbl_siteips.locationId =
            med_providers.bindtolocationid " &_
            "WHERE remote_addr LIKE '" & GetSiteAddress & "%' " &_
            "ORDER BY med_providers.name, med_providers.address, med_providers.cityandstate;"

        dim sql
        sql = "" &_
            "SELECT tbl_users.addressID, tbl_users.userName, tbl_users.userid,
            tbl_applications.*, tbl_w4.*, " &_
                "tbl_addresses.address AS mainAddress, tbl_addresses.addressTwo AS
                mainAddressTwo, " &_
                "tbl_addresses.city as mainCity, tbl_addresses.state AS mainState,
                tbl_addresses.zip AS mainZip " &_
            "FROM ((tbl_users RIGHT JOIN tbl_applications ON tbl_users.applicationID =
            tbl_applications.applicationID) " &_
                "LEFT JOIN tbl_w4 ON tbl_users.userID = tbl_w4.userid) " &_
                "LEFT JOIN tbl_addresses ON tbl_users.addressID = tbl_addresses.addressID " &_
            "WHERE (((tbl_applications.applicationID)=" & Me.ApplicationId & "));"

        LoadApplication = LoadApplicationData(sql, sqlMedicalProviders)

    end function

    public function AddEnrollment(siteId, applicantId)
        dim Enrollment

        if not isnull(applicantId) then
            set Enrollment = new cEnrollment
            with Enrollment
                .SiteId = siteId
                .ApplicantId = applicantId
            end with
            m_Enrollments.Add Enrollment.SiteId, Enrollment
            set Enrollment = nothing
        end if

    end function

    public function LoadOldEnrollment()
        dim sql
        sql = "" &_
            "SELECT " &_
                "LastnameFirst, " &_
                "Address, " &_
                "City, " &_
                "State, " &_
                "Zip, " &_
                "ApplicantStatus, " &_
                "Telephone, " &_
                "Applicants.[2ndTelephone] AS AltTelephone, " &_
                "ShortMemo, " &_
                "DateAvailable, " &_
                "k AS Skills, " &_
                "AppChangedBy, " &_
                "AppChangedDate, " &_
                "EmailAddress, " &_
                "TaxJurisdiction, " &_
                "MaritalStatus " &_
            "FROM Applicants " &_
            "WHERE ApplicantID=" & me.ApplicationId

        LoadOldEnrollment = LoadOldEnrollmentData (sql)
    end function

'#############  Private Functions ##############

    private function PCase(strInput)
        Dim currentPosition
        Dim nextSpaceAfter
        Dim outputString

        currentPosition = 1
        do while InStr(currentPosition, strInput, " ", 1) <> 0
            nextSpaceAfter = InStr(currentPosition, strInput, " ", 1)
            outputString = outputString &_
                UCase(Mid(strInput, currentPosition, 1)) &_
                LCase(Mid(strInput, currentPosition + 1, nextSpaceAfter - currentPosition))
                currentPosition = nextSpaceAfter + 1
            loop
        outputString = outputString &_
            UCase(Mid(strInput, currentPosition, 1)) &_
            LCase(Mid(strInput, currentPosition + 1))

        PCase = outputString

    end function

    private function StripCharacters (TextString)
        Dim RegularExpression

        If Len(TextString) > 0 Then
            Set RegularExpression = New RegExp
            RegularExpression.Pattern = "[<>']"
            RegularExpression.Global = True
            If VarType(TextString) = 8 Then
                StripCharacters = RegularExpression.Replace(TextString, "")
            End If
        End If

    end function


    private function FormatPhoneNumber(p_PhoneNumber)

        if len(p_PhoneNumber) > 0 then
            dim re

            set re = New RegExp
            with re
                .Pattern = "[()-.<>'$]"
                .Global = True
            end with

            FormatPhoneNumber = re.Replace(p_PhoneNumber, "")
        end if

    end function

    private function FillApplicationFromRS(p_RS)

       if not p_RS.eof then
            Me.Address                  = p_RS.fields("mainAddress").Value
            Me.City                     = p_RS.fields("mainCity").Value
            Me.State                    = p_RS.fields("mainState").Value
            Me.Zip                      = p_RS.fields("mainZip").Value
            if isnull(Me.Address) then
                Me.Address                  = p_RS.fields("addressOne").Value
                Me.City                     = p_RS.fields("city").Value
                Me.State                    = p_RS.fields("appState").Value
                Me.Zip                      = p_RS.fields("zipcode").Value
            end if
            'Me.ApplicantStatus         = p_RS.fields("ApplicantStatus").Value
            Me.Telephone                = p_RS.fields("mainPhone").Value
            Me.AltTelephone             = p_RS.fields("altPhone").Value
            'Me.ShortMemo               = p_RS.fields("ShortMemo").Value
            'Me.DateAvailable           = p_RS.fields("DateAvailable").Value
            Me.Skills                   = p_RS.fields("skillsSet").Value
            'Me.AppChangedBy            = p_RS.fields("AppChangedBy").Value
            'Me.AppChangedDate          = p_RS.fields("AppChangedDate").Value
            Me.EmailAddress             = p_RS.fields("email").Value
            'Me.TaxJurisdiction         = p_RS.fields("TaxJurisdiction").Value
            Me.MaritalStatus            = p_RS.fields("maritalStatus").Value
            Me.DOB                      = p_RS.fields("dob").Value
            Me.AliasNames               = p_RS.fields("aliasNames").Value
            Me.SSN                      = p_RS.fields("ssn").Value
            Me.Lastname                 = p_RS.fields("lastName").Value
            Me.Firstname                = p_RS.fields("firstName").Value
            Me.EmailUpdates             = p_RS.fields("emailupdates").Value
            Me.DesiredWageAmount        = p_RS.fields("desiredWageAmount").Value
            Me.MinWageAmount            = p_RS.fields("minWageAmount").Value
            Me.CurrentlyEmployed        = p_RS.fields("currentlyEmployed").Value
            Me.Sex                      = p_RS.fields("sex").Value
            Me.Smoker                   = p_RS.fields("smoker").Value
            Me.Citizen                  = p_RS.fields("citizen").Value
            Me.AlienType                = p_RS.fields("alienType").Value
            Me.AlienNumber              = p_RS.fields("alienNumber").Value
            Me.WorkAuthProof            = p_RS.fields("workAuthProof").Value
            Me.WorkTypeDesired          = StripCharacters(p_RS.fields("workTypeDesired").Value)
            Me.WorkAge                  = p_RS.fields("workAge").Value
            Me.WorkValidLicense         = p_RS.fields("workValidLicense").Value
            Me.WorkLicenseType          = p_RS.fields("workLicenseType").Value
            Me.AutoInsurance            = p_RS.fields("autoInsurance").Value
            Me.WorkRelocate             = StripCharacters(p_RS.fields("workRelocate").Value)
            Me.WorkConviction           = StripCharacters(p_RS.fields("workConviction").Value)
            Me.WorkConvictionExplain    = p_RS.fields("workConvictionExplain").Value
            Me.ECFullName               = p_RS.fields("ecFullName").Value
            Me.ECAddress                = p_RS.fields("ecAddress").Value
            Me.ECPhone                  = p_RS.fields("ecPhone").Value
            Me.ECDoctor                 = p_RS.fields("ecDoctor").Value
            Me.ECDocPhone               = p_RS.fields("ecDocPhone").Value
            Me.W4a                      = p_RS.fields("a").Value
            Me.W4b                      = p_RS.fields("b").Value
            Me.W4c                      = p_RS.fields("c").Value
            Me.W4d                      = p_RS.fields("d").Value
            Me.W4e                      = p_RS.fields("e").Value
            Me.W4f                      = p_RS.fields("f").Value
            Me.W4g                      = p_RS.fields("g").Value
            'Me.W4h                      ='sum of W4 collection
            Me.W4more                   = p_RS.fields("more").Value
            Me.MiddleInit               = p_RS.fields("middleinit").Value
            Me.W4Total                  = p_RS.fields("total").Value
            Me.W4Exempt                 = p_RS.fields("exempt").Value
            Me.W4Filing                 = p_RS.fields("filing").Value
            Me.W4nameDiffers            = p_RS.fields("namediffers").Value
            Me.AvailableWhen            = p_RS.fields("availableWhen").Value
            Me.WorkCommuteHow           = p_RS.fields("workCommuteHow").Value
            Me.WorkCommuteDistance      = p_RS.fields("workCommuteDistance").Value
            Me.WorkLicenseState         = p_RS.fields("workLicenseState").Value
            Me.WorkLicenseExpire        = p_RS.fields("workLicenseExpire").Value
            Me.WorkLicenseNumber        = p_RS.fields("workLicenseNumber").Value
            Me.AutoInsurance            = p_RS.fields("autoInsurance").Value
            Me.EduLevel                 = p_RS.fields("eduLevel").Value
            Me.AdditionalInfo           = p_RS.fields("additionalInfo").Value
            Me.SkillsSet                = p_RS.fields("skillsSet").Value
            Me.EmployerNameHistOne      = p_RS.fields("employerNameHistOne").Value
            Me.JobHistAddOne            = p_RS.fields("jobHistAddOne").Value
            Me.JobHistPhoneOne          = p_RS.fields("jobHistPhoneOne").Value
            Me.JobHistPayOne            = p_RS.fields("jobHistPayOne").Value
            Me.JobHistCityStateZipOne   = StripCharacters(p_RS.fields("jobHistCityOne").Value)
            & ", " & StripCharacters(p_RS.fields("jobHistStateOne").Value) & " " &
            StripCharacters(p_RS.fields("jobHistZipOne").Value)
            Me.JobHistSupervisorOne     = p_RS.fields("jobHistSupervisorOne").Value
            Me.JobDutiesOne             = p_RS.fields("jobDutiesOne").Value
            Me.JobHistFromDateOne       = p_RS.fields("jobHistFromDateOne").Value
            Me.JobHistToDateOne         = p_RS.fields("JobHistToDateOne").Value
            Me.JobReasonOne                 = p_RS.fields("jobReasonOne").Value
            Me.EmployerNameHistTwo          = p_RS.fields("employerNameHistTwo").Value
            Me.JobHistAddTwo                = p_RS.fields("jobHistAddTwo").Value
            Me.JobHistPhoneTwo              = p_RS.fields("jobHistPhoneTwo").Value
            Me.JobHistPayTwo                = p_RS.fields("jobHistPayTwo").Value
            Me.JobHistCityStateZipTwo       = StripCharacters(p_RS.fields("jobHistCityTwo").Value)
            & ", " & StripCharacters(p_RS.fields("jobHistStateTwo").Value) & " " &
            StripCharacters(p_RS.fields("jobHistZipTwo").Value)
            Me.JobHistSupervisorTwo         = p_RS.fields("jobHistSupervisorTwo").Value
            Me.JobDutiesTwo                 = p_RS.fields("jobDutiesTwo").Value
            Me.JobHistFromDateTwo           = p_RS.fields("jobHistFromDateTwo").Value
            Me.JobHistToDateTwo             = p_RS.fields("JobHistToDateTwo").Value
            Me.JobReasonTwo                 = p_RS.fields("jobReasonTwo").Value
            Me.EmployerNameHistThree        = PCase(StripCharacters(p_RS.fields(
            "employerNameHistThree").Value))
            Me.JobHistAddThree              = StripCharacters(p_RS.fields("jobHistAddThree").Value)
            Me.JobHistPhoneThree            = StripCharacters(p_RS.fields("jobHistPhoneThree").
            Value)
            Me.JobHistPayThree              = StripCharacters(p_RS.fields("jobHistPayThree").Value)
            Me.JobHistCityStateZipThree = StripCharacters(p_RS.fields("jobHistCityThree").Value
            ) & ", " & StripCharacters(p_RS.fields("jobHistStateThree").Value) & " " &
            StripCharacters(p_RS.fields("jobHistZipThree").Value)
            Me.JobHistSupervisorThree       = StripCharacters(p_RS.fields("jobHistSupervisorThree"
            ).Value)
            Me.JobDutiesThree               = p_RS.fields("jobDutiesThree").Value
            Me.JobHistFromDateThree         = StripCharacters(p_RS.fields("jobHistFromDateThree").
            Value)
            Me.JobHistToDateThree           = StripCharacters(p_RS.fields("JobHistToDateThree").
            Value)
            Me.JobReasonThree               = p_RS.fields("jobReasonThree").Value
            Me.PAndPAgree                   = p_RS.fields("pandpAgree").Value
            Me.UnEmpAgree                   = p_RS.fields("unempAgree").Value
            Me.ApplicantAgree               = p_RS.fields("applicantAgree").Value
            Me.UserName                     = p_RS.fields("userName").Value
            Me.UserId                       = p_RS.fields("userid").Value
            Me.DateLastEnrolled             = p_RS.fields("lastInserted").Value
            Me.DateLastModified             = p_RS.fields("modifiedDate").Value
            Me.DateCreated                  = p_RS.fields("creationDate").Value
            Me.AddEnrollment PER, p_RS.fields("inPER").Value
            Me.AddEnrollment BUR, p_RS.fields("inBUR").Value
            Me.AddEnrollment BOI, p_RS.fields("inBOI").Value
            Me.AddEnrollment IDA, p_RS.fields("inIDA").Value
            Me.AddEnrollment PPI, p_RS.fields("inPPI").Value
            Me.AddEnrollment POC, p_RS.fields("inPOC").Value
        end if
    end function

    public function FillMedicalProvidersFromRS(p_RS)
        dim MedicalProvider, iCount

        if not p_RS.eof then m_WhereWeAre = p_RS.fields("site").Value

        do while not ( p_RS.eof )
            iCount = iCount + 1

            set MedicalProvider = New cMedicalProvider
            with MedicalProvider
                .ProviderId = p_RS.fields("id").Value
                .Name       = p_RS.fields("name").Value
                .Address    = p_RS.fields("address").Value
                .Cityline   = p_RS.fields("cityandstate").Value
                .Phone      = p_RS.fields("phone").Value
                .Fax        = p_RS.fields("fax").Value
                .HoursOpen  = p_RS.fields("openhours").Value
            end with

            m_MedicalProviders.Add iCount, MedicalProvider
            p_RS.MoveNext
        loop
    end function

    private function FillOldEnrollmentFromRS(p_RS)
        with Me
            .LastnameFirst            = p_RS.fields("LastnameFirst").Value
            .Address                  = p_RS.fields("Address").Value
            .City                     = p_RS.fields("City").Value
            .State                    = p_RS.fields("State").Value
            .Zip                      = p_RS.fields("Zip").Value
            .ApplicantStatus          = p_RS.fields("ApplicantStatus").Value
            .Telephone                = p_RS.fields("Telephone").Value
            .AltTelephone             = p_RS.fields("AltTelephone").Value
            .ShortMemo                = p_RS.fields("ShortMemo").Value
            .DateAvailable            = p_RS.fields("DateAvailable").Value
            .Skills                   = p_RS.fields("Skills").Value
            .AppChangedBy             = p_RS.fields("AppChangedBy").Value
            .AppChangedDate           = p_RS.fields("AppChangedDate").Value
            .EmailAddress             = p_RS.fields("EmailAddress").Value
            .TaxJurisdiction          = p_RS.fields("TaxJurisdiction").Value
            .MaritalStatus            = p_RS.fields("MaritalStatus").Value
        end with
    end function

    private function LoadOldEnrollmentData(sql)
        dim rs
        set rs = GetRSfromDB(sql, dsnLessTemps(getTempsDSN(me.SiteDSN)))
        FillOldEnrollmentFromRS(rs)
        LoadTempsData = rs.recordcount
        rs. close
        set rs = nothing
    end function

   private function LoadApplicationData(sql, sqlMedicalProviders)

        dim rs
        set rs = GetRSfromDB(sql, MySql)
        FillApplicationFromRS(rs)

        set rs = GetRSfromDB(sqlMedicalProviders, MySql)
        FillMedicalProvidersFromRS(rs)

        LoadApplicationData = rs.recordcount
        rs.close
        set rs = nothing
    end function

end class


class cEnrollment
    private m_SiteId
    private m_CompCode
    private m_ApplicantId

    'Applicant
    private m_LastnameFirst
    private m_Address
    private m_City
    private m_State
    private m_Zip
    private m_ApplicantStatus
    private m_Telephone
    private m_AltTelephone
    private m_ShortMemo 'Active'
    private m_DateAvailable
    private m_k 'CurrentApplication.SkillsSet
    private m_AppChangedBy
    private m_AppChangedDate
    private m_EmailAddress
    private m_TaxJurisdiction
    'private m_MaritalStatus 'UCase

    'PR3MSTR
    private m_DateHired
    private m_Birthdate
    private m_MaritalStatus 'Ucase MaritalStatus
    private m_EmpChangedBy 'V-user_id
    private m_EmpChangedDate
    private m_StateExemptions '.W4Total)
    private m_FedExemptions '.W4Total
    private m_TaxJurisdictionTaxPackageId 'Default = 3


    public property get SiteId
        SiteId = m_SiteId
    end property
    public property let SiteId(p_SiteId)
        m_SiteId = p_SiteId
        m_CompCode = getTempsCompCode(p_SiteId)
    end property

    public property get CompCode()
        CompCode = m_CompCode
    end property

    public property get ApplicantId
        ApplicantId = m_ApplicantId
    end property
    public property let ApplicantId(p_ApplicantId)
        m_ApplicantId = p_ApplicantId
    end property

        'Applicant

    public property get LastnameFirst
        LastnameFirst = m_LastnameFirst
    end property
    public property let LastnameFirst(p_LastnameFirst)
        m_LastnameFirst = p_LastnameFirst
    end property


    public property get Address
        Address = m_Address
    end property
    public property let Address(p_Address)
        m_Address = p_Address
    end property
    public property get City
        City = m_City
    end property
    public property let City(p_City)
        m_City = p_City
    end property


    public property get State
        State = m_State
    end property
    public property let State(p_State)
        m_State = p_State
    end property


    public property get Zip
        Zip = m_Zip
    end property
    public property let Zip(p_Zip)
        m_Zip = p_Zip
    end property


    public property get ApplicantStatus
        ApplicantStatus = m_ApplicantStatus
    end property
    public property let ApplicantStatus(p_ApplicantStatus)
        m_ApplicantStatus = p_ApplicantStatus
    end property


    public property get Telephone
        Telephone = m_Telephone
    end property
    public property let Telephone(p_Telephone)
        m_Telephone = p_Telephone
    end property


    public property get AltTelephone
        AltTelephone = m_AltTelephone
    end property
    public property let AltTelephone(p_AltTelephone)
        m_AltTelephone = p_AltTelephone
    end property

    'Active'
    public property get ShortMemo
        ShortMemo = m_ShortMemo
    end property
    public property let ShortMemo(p_ShortMemo)
        m_ShortMemo = p_ShortMemo
    end property

    public property get DateAvailable
        DateAvailable = m_DateAvailable
    end property
    public property let DateAvailable(p_DateAvailable)
        m_DateAvailable = p_DateAvailable
    end property

    'CurrentApplication.SkillsSet
    public property get k
        k = m_k
    end property
    public property let k(p_k)
        m_k = p_k
    end property

    public property get AppChangedBy
        AppChangedBy = m_AppChangedBy
    end property
    public property let AppChangedBy(p_AppChangedBy)
        m_AppChangedBy = p_AppChangedBy
    end property

    public property get AppChangedDate
        AppChangedDate = m_AppChangedDate
    end property
    public property let AppChangedDate(p_AppChangedDate)
        m_AppChangedDate = p_AppChangedDate
    end property

    public property get EmailAddress
        EmailAddress = m_EmailAddress
    end property
    public property let EmailAddress(p_EmailAddress)
        m_EmailAddress = p_EmailAddress
    end property

    public property get TaxJurisdiction
        TaxJurisdiction = m_TaxJurisdiction
    end property
    public property let TaxJurisdiction(p_TaxJurisdiction)
        m_TaxJurisdiction = p_TaxJurisdiction
    end property

    'PR3MSTR

    public property get DateHired
        DateHired = m_DateHired
    end property
    public property let DateHired(p_DateHired)
        m_DateHired = p_DateHired
    end property

    public property get Birthdate
        Birthdate = m_Birthdate
    end property
    public property let Birthdate(p_Birthdate)
        m_Birthdate = p_Birthdate
    end property

    public property get MaritalStatus
        MaritalStatus = m_MaritalStatus
    end property
    public property let MaritalStatus(p_MaritalStatus)

        if not isnull(p_MaritalStatus) then
            if len(p_MaritalStatus) > 0 then
                m_MaritalStatus = ucase(p_MaritalStatus)
            end if
        end if

    end property
    'V-user_id
    public property get EmpChangedBy
        EmpChangedBy = m_EmpChangedBy
    end property
    public property let EmpChangedBy(p_EmpChangedBy)
        m_EmpChangedBy = p_EmpChangedBy
    end property

    public property get EmpChangedDate
        EmpChangedDate = m_EmpChangedDate
    end property
    public property let EmpChangedDate(p_EmpChangedDate)
        m_EmpChangedDate = p_EmpChangedDate
    end property

    '.W4Total
    public property get StateExemptions
        StateExemptions = m_StateExemptions
    end property
    public property let StateExemptions(p_StateExemptions)
        m_StateExemptions = p_StateExemptions
    end property

    '.W4Total
    public property get FedExemptions
        FedExemptions = m_FedExemptions
    end property
    public property let FedExemptions(p_FedExemptions)
        m_FedExemptions = p_FedExemptions
    end property

    'Default = 3
    public property get TaxJurisdictionTaxPackageId
        TaxJurisdictionTaxPackageId = m_TaxJurisdictionTaxPackageId
    end property
    public property let TaxJurisdictionTaxPackageId(p_TaxJurisdictionTaxPackageId)
        m_TaxJurisdictionTaxPackageId = p_TaxJurisdictionTaxPackageId
    end property


'########################## Public Functions #############################
    public function LoadNewEnrollmentData(p_Application)

        with me
            'Applicant
            .ApplicantId                = p_Application.ApplicantId
            .LastnameFirst              = p_Application.LastnameFirst
            .Address                    = p_Application.Address
            .City                       = p_Application.City
            .State                      = p_Application.State
            .Zip                        = p_Application.Zip
            .ApplicantStatus            = 1
            .Telephone                  = p_Application.Telephone
            .AltTelephone               = p_Application.AltTelephone
            .ShortMemo                  = "Active"
            .DateAvailable              = p_Application.DateLastEnrolled
            .k                          = p_Application.SkillsSet
            .AppChangedBy               = "V-" & cstr(p_Application.UserId)
            .AppChangedDate             = p_Application.DateLastModified
            .EmailAddress               = p_Application.EmailAddress
            .TaxJurisdiction            = "ID"
            .MaritalStatus              = p_Application.MaritalStatus
            'PR3MSTR
            .DateHired                  = p_Application.ApplicantAgree
            .Birthdate                  = p_Application.DOB
            .MaritalStatus              = p_Application.MaritalStatus
            '.EmpChangedBy               = p_Application.EmpChangedBy
            '.EmpChangedDate             = p_Application.EmpChangedDate
            .StateExemptions            = p_Application.W4total
            .FedExemptions              = p_Application.W4total
            '.TaxJurisdictionTaxPackageId = p_Application.TaxJurisdictionTaxPackageId
        end with

    end function

    public function LoadExistingEnrollment()

        dim sql
        sql = "" &_

        "SELECT LastnameFirst, Address, City, State, Zip, ApplicantStatus, " &_
        "Applicants.Telephone, Applicants.[2ndTelephone], Applicants.ShortMemo, " &_
        "Applicants.DateAvailable, Applicants.k, Applicants.AppChangedBy, " &_
        "Applicants.AppChangedDate, Applicants.EmailAddress, " &_
        "Applicants.TaxJurisdiction AS 1stTaxJurisdiction, " &_
        "Applicants.MaritalStatus AS 1stMaritalStatus, PR3MSTR.* " &_
        "FROM Applicants LEFT JOIN PR3MSTR ON Applicants.ApplicantID = PR3MSTR.ApplicantId " &_
            "WHERE Applicants.[ApplicantID]=" & cstr(Me.ApplicantId) & ";"

        LoadExistingEnrollment = LoadExistingEnrollmentData(sql)

    end function


'########################## Private Functions ###########################

    private function FillEnrollmentData (p_RS)

        if not p_RS.eof then
            with me
                'Applicant
                .LastnameFirst              = p_RS.fields("LastnameFirst").Value
                .Address                    = p_RS.fields("Address").Value
                .City                       = p_RS.fields("City").Value
                .State                      = p_RS.fields("State").Value
                .Zip                        = p_RS.fields("Zip").Value
                .ApplicantStatus            = p_RS.fields("ApplicantStatus").Value
                .Telephone                  = p_RS.fields("Telephone").Value
                .AltTelephone               = p_RS.fields("2ndTelephone").Value
                .ShortMemo                  = p_RS.fields("ShortMemo").Value
                .DateAvailable              = p_RS.fields("DateAvailable").Value
                .k                          = p_RS.fields("k").Value
                .AppChangedBy               = p_RS.fields("AppChangedBy").Value
                .AppChangedDate             = p_RS.fields("AppChangedDate").Value
                .EmailAddress               = p_RS.fields("EmailAddress").Value
                .TaxJurisdiction            = p_RS.fields("1stTaxJurisdiction").Value
                .MaritalStatus              = p_RS.fields("1stMaritalStatus").Value 'UCase

                'PR3MSTR
                .DateHired                  = p_RS.fields("DateHired").Value
                .Birthdate                  = p_RS.fields("Birthdate").Value
                .MaritalStatus              = p_RS.fields("MaritalStatus").Value
                '.EmpChangedBy               = p_RS.fields("EmpChangedBy").Value
                '.EmpChangedDate             = p_RS.fields("EmpChangedDate").Value
                .StateExemptions            = p_RS.fields("StateExemptions").Value
                .FedExemptions              = p_RS.fields("FedExemptions").Value
                '.TaxJurisdictionTaxPackageId = p_RS.fields("TaxJurisdictionTaxPackageId").Value
            end with
        end if

    end function

    private function LoadExistingEnrollmentData(sql)
        dim rs

        set rs = GetRSfromDB(sql, dsnLessTemps(me.SiteId))
        FillEnrollmentData(rs)

        LoadExistingEnrollmentData = rs.recordcount
        rs.close
        set rs = nothing

    end function

end class

class cMedicalProvider
    private m_ProviderId
    private m_Name
    private m_Address
    private m_Cityline
    private m_Phone
    private m_Fax
    private m_HoursOpen

    public property get ProviderId()
        ProviderId = m_ProviderId
    end property
    public property let ProviderId(p_ProviderId)
        m_ProviderId = p_ProviderId
    end property

    public property get Name()
        Name = m_Name
    end property
    public property let Name(p_Name)
        m_Name = p_Name
    end property

    public property get Address()
        Address = m_Address
    end property
    public property let Address(p_Address)
        m_Address = p_Address
    end property

    public property get Cityline()
        Cityline = m_Cityline
    end property
    public property let Cityline(p_Cityline)
        m_Cityline = p_Cityline
    end property

    public property get Phone()
        Phone = m_Phone
    end property
    public property let Phone(p_Phone)
```

```
        m_Phone = p_Phone
    end property

    public property get Fax()
        Fax = m_Fax
    end property
    public property let Fax(p_Fax)
        m_Fax = p_Fax
    end property

    public property get HoursOpen()
        HoursOpen = m_HoursOpen
    end property
    public property let HoursOpen(p_HoursOpen)
        m_HoursOpen = p_HoursOpen
    end property

end class

%>
```