**PANTALLA: PRECALIFICACIÓN -> BTNENVIAR**

**Objetos**

* BandejaPreSolicitud.aspx
* Precalificacion.aspx
* Precalificacion.aspx.cs

Entidad -> Producto oProducto = new Producto();

Listado -> List<SeccionPrecalificacion> oSeccion = new List<SeccionPrecalificacion>();

Toma el producto seleccionado, y envía para obtener oseccion

oSeccion = \_presenter.onSeccionPrecal(oProducto);

tipo de formulario lo obtiene del campo “TABCARGUME” del oSeccion

TipoFormularioTmp = oSeccion[0].TABCARGUME.ToString().Trim();

Después valida si el registro es nuevo o si ya existe,

Si es nuevo (lblNumeroSolicitud.Text.Trim() == "0")

Valida datos

Flag = true;

ValidacionRuc = true;

//Préstamo Personal setea todos en color rojo

* Tipo Empleo
* Tipo Evaluación
* Moneda
* Titular
* Monto
* Desgravamen
* Plazo
* Tasa
* Sub Producto
* Tipo Cuota Doble
* Código Convenio
* Descripción Convenio
* Tipo Convenio Descripción
* Ruc Convenio
* Fecha Control (Primer Vencimiento)

**Valida, si esta en 0(cero, seleccionado) es persona natural, y realiza la siguiente validación**

Setea colores en negro

* txtDocIdentidadSol.BorderColor =
* txtApellidoPaternoSol.BorderColor =
* txtApellidoMaternoSol.BorderColor =
* txtPrimerNombreSol.BorderColor =
* txtDocIdentidadCon.BorderColor =
* txtApellidoPaternoCon.BorderColor =
* txtApellidoMaternoCon.BorderColor =
* txtPrimerNombreCon.BorderColor =
* txtRUCLab.BorderColor =
* txtRazonSocialLab.BorderColor =
* txtCorreo.BorderColor =
* UCFechaNacimiento.txtFechaControl.BorderColor =
* ddlProducto.ForeColor =
* ddlDocIdentidadSol.ForeColor =
* ddlDocIdentidadCon.ForeColor =
* System.Drawing.Color.Empty;

Comienza a validar campos

* Producto no está seleccionado, setea color a rojo
* Documento identidad Solicitante es igual a la lista de tipo de documentos (ruc), setea color en rojo
* documento identidad cónyuge es igual a la lista de tipo de documentos(ruc) y el número de documento es mayor que uno setea color a rojo

Validación solicitante

* Documento identidad solicitante
* Apellido paterno solicitante
* Apellido materno solicitante
* Primer Nombre solicitante
* Apellido paterno solicitante
* Apellido materno solicitante
* Primer Nombre solicitante
* DNI Solicitante
* Validación cónyuge
* Documento identidad cónyuge
* Apellido paterno cónyuge
* Apellido materno cónyuge
* Primer Nombre cónyuge
* Apellido paterno cónyuge
* Apellido materno cónyuge
* Primer Nombre cónyuge
* DNI Cónyuge

validación de los integrantes naturales

* Numero documento identidad, apellido paterno, apellido materno, primero nombre no tenga caracteres menos que 2.
* Apellidos paterno, materno, primero nombre tenga texto sin número.
* Tipo de documento de identidad sea igual a entidad “TipoDocumentoIdentidadEnum.DNI”

validación de los integrantes jurídicos(Aval)

* Numero de documento identidad su longitud sea mayor que 3
* Texto de razón social su longitud sea mayor que 2
* Numero documento identidad validad si tiene los 11 dígitos

validación de laboral

* Ingreso Ruc del ruc longitud sea mayor que 3
* Texto de razón social su longitud sea mayor que 2
* Numero documento identidad validad si tiene los 11 dígitos

validación solo hipotecarios (SeccionPanelTmp = 7)

* Correo valido
* Fecha de nacimiento no sea igual a 01/01/0001

Validación de Datos para Préstamo Personal (SeccionPanelTmp = 4)

* Si ValidacionRuc sigue estando en true, valida que ruc laboral tenga 11 dígitos
* Razón social laboral su longitud no sea menor que 2
* Correo valido
* Fecha de nacimiento no sea igual a 01/01/0001

**Valida, si está en 1(cero, seleccionado) es persona jurídica, y realiza la siguiente validación**

Setea colores en negro

* txtNroDocIdentidad.BorderColor =
* txtPrimerNombre.BorderColor =
* txtApellidoPaterno.BorderColor =
* txtApellidoMaterno.BorderColor =
* System.Drawing.Color.Empty;
* Numero documento identidad sea mayor que 3
* Apellido paterno, materno, primer nombre su longitud no sea menor que 2
* Apellido paterno, materno y primero nombre no obtenga ningún numero
* Tipo documento identidad sea DNI.

validación de los integrantes jurídicos

* Numero documento identidad sea mayor que 3
* Razón social su longitud no sea menor que 2
* Numero de documento identidad su longitud sea de 11 dígitos

**Una vez validado los campos realiza el llamado a la siguiente función “OnbtnGuardarNuevo\_Click” enviando los campos del solicitante.**

->Invoca al objeto “SolcredController\_Precalificacion.cs” a la función “Precalificacion\_Insertar(datos solicitante)”

->Invoca al objeto”SolicitudPre.cs” que se encuentra en el Back-end a la función “InsertarPrecal(datos solicitante)” que es un cobol STSP130

->Invoca al objeto “ProgramSTSP130” a la función “InsertarSTSP130(datos solicitantes)”

* Inserta datos enviando datos al cobol “STSP130”
* Envia datos “codigosemento, código producto, código sub producto, tipo evaluacion” a la funcion “TipoEvaluacionDAO.ObtenerFlagDxP(oTipEval)” para obtener si el “FLGCDXP = 1” genera el generarDXP
* request1.TipoFormulario = “07” || oSolicitudResultado.NumeroSolicitud != 0

Grabar Datos-Adicionales Panel 7 de los datos de solicitante

InsertarSTSP132(datos solicitantes)

Grabar Datos Hipotecario + Cash

InsertarSTSP134(datos solicitantes)

* Obtiene datos del cobol “oSolicitud”
* Valida si “oSolicitud.NumeroSolicitud != 0 && oSolicitante.ObtenerConvenio.TipoBusqueda != 0”

Invoca al objeto “ProgramSTSP914” a la función “guardarSTSP914(solicitante. ObtenerConvenio, del resultado envía NumeroSolicitud)”

* Valida si oSolicitud.NumeroSolicitud != 0

Guarda número de solicitud

Datos del solicitante obtiene CodigoOpcion = 2, NumeroSolicitudSTS, UltimoNombreBusqueda, Integrantes [20,array] y los guarda en “oSolicitante”

Invoca al objeto “solcredController\_Precalifiacion.cs” la función “controller.Precalificacion\_Seleccionar(oSolicitante)” el cual también invoca al objeto “ISolicitudPre.cs” y este invoca al objeto “SolicitudPre.cs”a la función “SeleccionarPrecal(oSolicitante)” el cual relaliza la siguiente acción

Invoca al objeto “ProgramSTSP130.cs” a la función “SeleccionarSTSP130(envía datos del solicitante)” “?” …obtiene el resultado de la ejecución “oSolicitudResultado” entidad “SolicitudPreEdicion”

De la respuesta evalúa si el producto es diferente de 0

“oSolicitudResultado.CodigoProducto != 0”, si es asi replaza los siguiente datos “request1.accion = 2, request1.nroPrecalificacion = request1.NumeroSolicitudSTS” y realiza un foreach realizando la llamada al objeto “ProgramSTSP132” a la función SeleccionarSTSP132(request1)

Recorre cada vuelta reemplazando los siguientes campos:

oSolicitudResultado.Modalidad = STSP132Resul.Modalidad;

oSolicitudResultado.ValorSeguro = STSP132Resul.ValorSeguro;

oSolicitudResultado.SeguroBien = STSP132Resul.SeguroBien;

oSolicitudResultado.PorcentajeLTV = STSP132Resul.PorcentajeLTV;

oSolicitudResultado.ImporteBien = STSP132Resul.ImporteBien;

oSolicitudResultado.FlagGastosH = STSP132Resul.FlagGastosH;

oSolicitudResultado.ImporteCuotaIni = STSP132Resul.ImporteCuotaIni;

oSolicitudResultado.PorcentajeCuotaIni = STSP132Resul.PorcentajeCuotaIni;

oSolicitudResultado.PorcLTVDefault = STSP132Resul.PorcLTVDefault;

oSolicitudResultado.Gastos01 = STSP132Resul.Gastos01;

oSolicitudResultado.Gastos02 = STSP132Resul.Gastos02;

oSolicitudResultado.Gastos03 = STSP132Resul.Gastos03;

oSolicitudResultado.Gastos04 = STSP132Resul.Gastos04;

oSolicitudResultado.ImporteGastos = STSP132Resul.ImporteGastos;

oSolicitudResultado.ValorDesgravamen = STSP132Resul.ValorDesgravamen;

oSolicitudResultado.TipoGarantia = STSP132Resul.TipoGarantia;

oSolicitudResultado.DetalleGarantia = STSP132Resul.DetalleGarantia;

oSolicitudResultado.TipoSeguro = STSP132Resul.TipoSeguro;

oSolicitudResultado.CapaSeguro = STSP132Resul.CapaSeguro;

oSolicitudResultado.PlanSeguro = STSP132Resul.PlanSeguro;

oSolicitudResultado.TCEA = STSP132Resul.TCEA;

oSolicitudResultado.Comision = STSP132Resul.Comision;

oSolicitudResultado.valorCDXP = STSP132Resul.valorCDXP;

oSolicitudResultado.resultadoCDXP = STSP132Resul.resultadoCDXP;

oSolicitudResultado.LimiteMaxCredSoles =STSP132Resul.LimiteMaxCredSoles;

oSolicitudResultado.LimiteMaxCredDolares = STSP132Resul.LimiteMaxCredDolares;

oSolicitudResultado.CreditoSugeridoSoles = STSP132Resul.CreditoSugeridoSoles;

oSolicitudResultado.CreditoSugeridoDolares = STSP132Resul.CreditoSugeridoDolares;

Terminado el foreach, crea la entidad oProducto “com.scotiabank.sts.business.entities.solcred.Producto”

Obtiene el producto “oSolicitudResultado.CodigoProducto” y lo reemplaza “oProducto.CodigoProducto”, realiza el llamado a la funcion “ListaSeccionPrecal( enviando la entidad oProducto)”, el cual realiza un llamado al objeto “SeccionPrecal(oProducto)” el cual realiza el llamado al store “usp\_seccionesPrecalificacion” enviando los parámetros “@TABCTABLA=775, @PRDCCODPRD= oProducto.CodigoProducto” el cual regresa una lista de la entidad “SeccionPrecalificacion.cs”

Del listado que vuelve el store los reemplaza en “request1.TipoFormulario”

Luego evalúa si el “request1.TipoFormulario es vacío iguala a 0” sino convierte a numérico y lo reemplaza “oSolicitudResultado.TipoFormulario”

Luego reemplaza “request1.accion = 2”

Luego realiza un foreach llamando al objeto “ProgramSTSP133” a la función “GenerarEscenarios133(enviando la entidad request1)” “?”, dentro del foreach crea la siguiente entidad STSP133Resul “entities.solcred.CuotasAproximadas”

, va guardando el resultado de cada ítem en “oSolicitudResultado.lstCuotaAproximadas.Add(STSP133Resul)”

Luego realiza un foreach llamando al objeto “ProgramSTSP134” a la función “SeleccionarSTSP134(request1)” “?”, dentro del foreach crea la entidad STSP134Resul “entities.solcred.SolicitantePre”, va guardando el resultado de cada ítem los siguiente campos

oSolicitudResultado.FlagHipotecarioCash = true;

oSolicitudResultado.ImporteCreditoCash = STSP134Resul.ImporteCreditoCash;

oSolicitudResultado.PlazoCash = STSP134Resul.PlazoCash;

oSolicitudResultado.TasaCash = STSP134Resul.TasaCash;

oSolicitudResultado.CodSeguroCash = STSP134Resul.CodSeguroCash;

oSolicitudResultado.FlagCuotaD = STSP134Resul.FlagCuotaD;

oSolicitudResultado.ImporteCNCash = STSP134Resul.ImporteCNCash;

oSolicitudResultado.ImporteCDCash = STSP134Resul.ImporteCDCash;

Por ultimo devuelve la entidad “oSolicitudResultado” a la función “Precalificacion\_Seleccionar” que se encuentra en el objeto “solcredController\_Precalificacion”.

Luego la respuesta lo recibe “oSolicitud”, luego obtiene de la entidad “oSolicitante” el numero solicitud sts para reemplazarlo en la entidad “oSolicitud.NumeroSolicitud”.

Luego se realiza una validación “oSolicitante.ObtenerConvenio.TipoBusqueda != 0”, si cumple crea la entidad objIn “ConvenioBusquedaPP” y reemplaza los siguiente campos:

objIn.TipoBusqueda = 5;

objIn.Precalificacion = oSolicitud.NumeroSolicitud;

objIn.UsuarioSTS = oSolicitante.UsuarioSTS;

Realiza un llamado al objeto “IConvenioBus.cs”, el cual también realiza el llamado al objeto “ConvenioBus.cs” a la función “ListarSTSP914”, el cual realiza el llamado al objeto “ProgramSTSP914.cs” a la función “ListarSTSP914()” “?”, el cual devuelve un listado y lo guarda en \_objOut “ConvenioBusquedaPP” y los datos obtenidos los lleva a la entidad “oSolicitud.ObtenerConvenio = \_objOut”.

Luego realiza un llamado “this.View.CargarDetalleSolicitud(oSolicitud)” el cual realiza un llamado al objeto “IPrecalificacionView.cs”

**Luego invoca a la función “EstablecerModoEdicionLectura()”**

**imgSemaforoGlobal.ImageUrl = imgResultadoSolicitud.ImageUrl;**

lblResGlobal.Text = lblResultadoEvaluacion.Text; lblNroSolGlobal.Text = lblNumeroSolicitud.Text;

**Luego invoca a la función “EvaluarResultadoSemaforo”, el cual realiza las validaciones cargando una lista de semáforos, si la lista viene con datos determina por la guía 954 si es que va a buro o no, si viene vacío entonces se comporta como siempre.**

**Luego muestra el popup “**ModalPopupBuro.Show();**”**

**Luego invoca a la función “**GuardarLiqGastos(lblNumeroSolicitud.Text);**” , el cual realiza que el producto sea diferente de los siguiente número “152,153,155,159,160”, si cumple con esas condiciones graba liquidación gastos en cobol “**STSP133**”.**