Brothersoft

Informe técnico aplicación Web

Contenido

[Arquitectura de la aplicación 2](#_Toc47431599)

[Diagrama de flujo 3](#_Toc47431600)

[Diccionario de datos 4](#_Toc47431601)

[WebServices 8](#_Toc47431602)

[Controladores 12](#_Toc47431603)

[Modelos 14](#_Toc47431604)

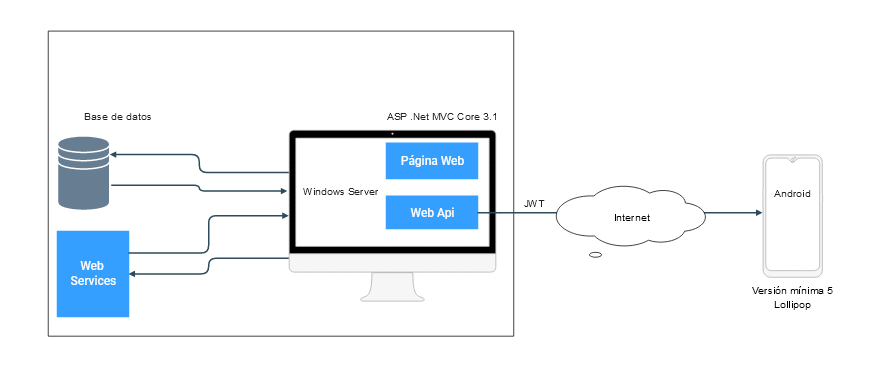
[Ayudas (Helper) 14](#_Toc47431605)

[Configuración 16](#_Toc47431606)

# Arquitectura de la aplicación

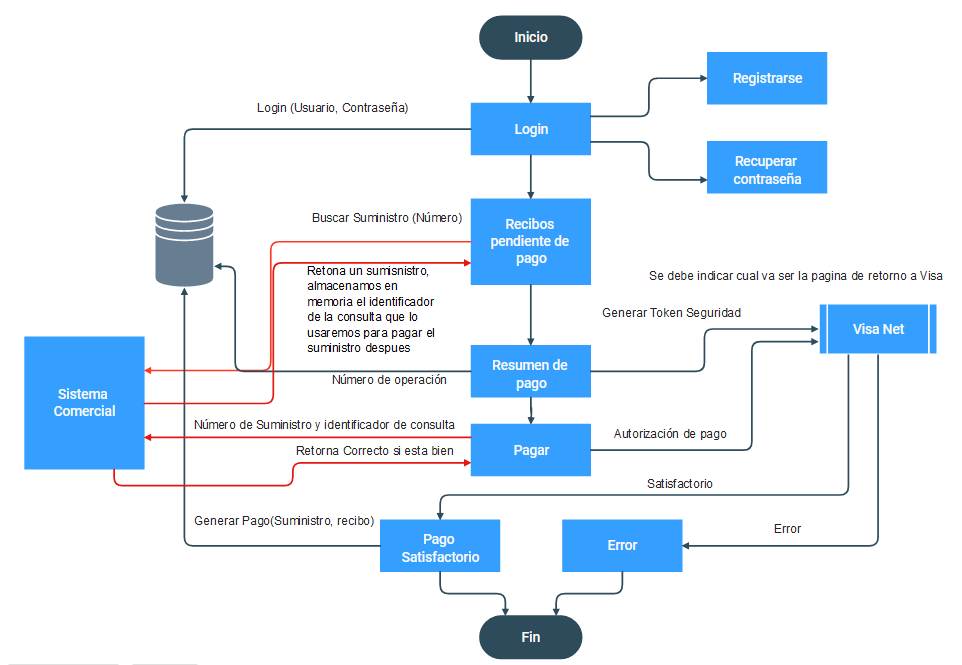
El desarrollo de aplicación esta dividida en 3 programas dos de ellos para la instalación en el servidor y el otro es un aplicativo móvil en Android.

* Página web
  + ASP Core 3.1
  + MVC
  + Sql Server
* WebApi
  + ASP Core 3.1
  + ASP MVC WebAPI
  + Sql Server
  + JWT
* Android
  + Versión mínima 5 Lollipop



# Diagrama de flujo

1. Para usar la aplicación el usuario debe registrarse que le envía un correo de confirmación de cuenta.
2. Cuando el usuario ingresa al sistema se le crea una sesión de 30 minutos.
3. Para buscar un suministro, se ingresa el numero de contrato, la aplicación genera un identificador (Guid) que se almacena en una sesión, con el cual consulta al webservices y este le retorna un suministro con su descripción y monto a pagar.
4. Si el usuario quiere pagar el suministro se le genera un numero de operación y se envía a Visa para pagar.
5. Si el pago es valido entonces almacenamos el pago y actualizamos en el sistema comercial con el identificador de la consulta y el numero de contrato.

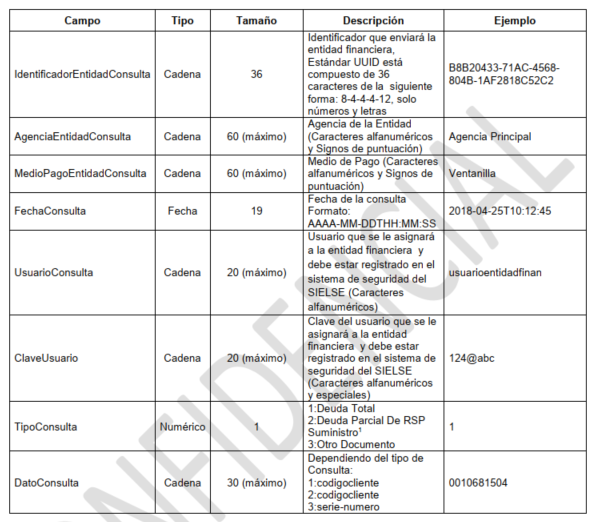


# WebServices Consulta de Deuda

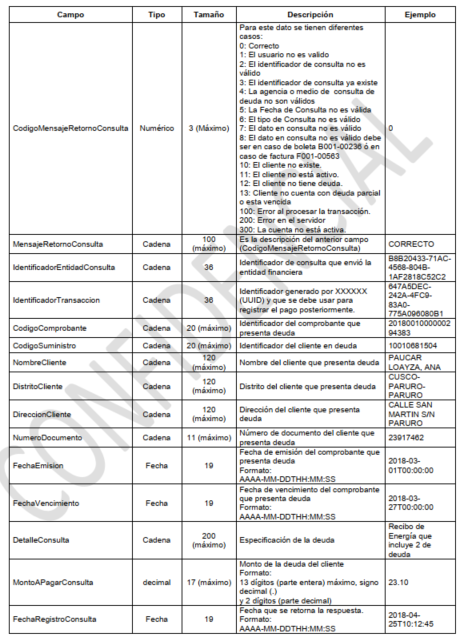
Este servicio se usa para determinar la deuda pendiente de un cliente. Para ello la entidad financiera enviará un conjunto de datos incluyendo el código del cliente acompañado por otros datos de seguridad, los cuales serán procesados por XXXXXX y éste devolverá un mensaje de respuesta.

Si la consulta es correcta (CodigoMensajeRetornoConsulta = 0), el mensaje de respuesta contendrá los datos del cliente, datos de seguridad y el monto a pagar, en caso contrario el mensaje de respuesta solo contendrá la descripción del error, dejando los demás campos sin valor (null).

Parámetros de consulta



Datos de respuesta

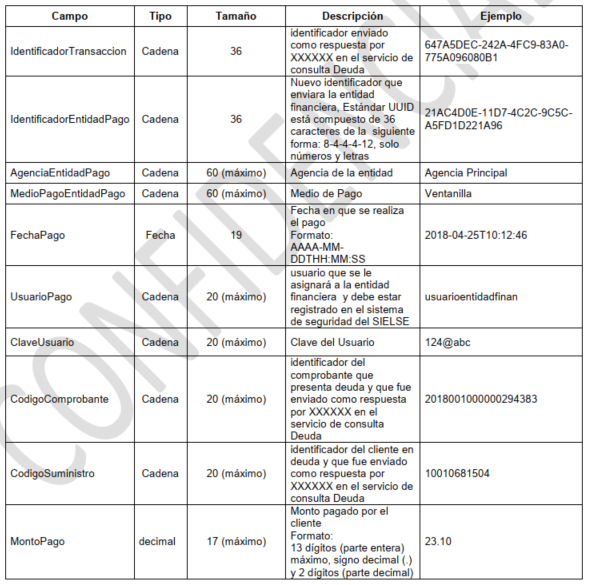


# WebServices PagoDeuda

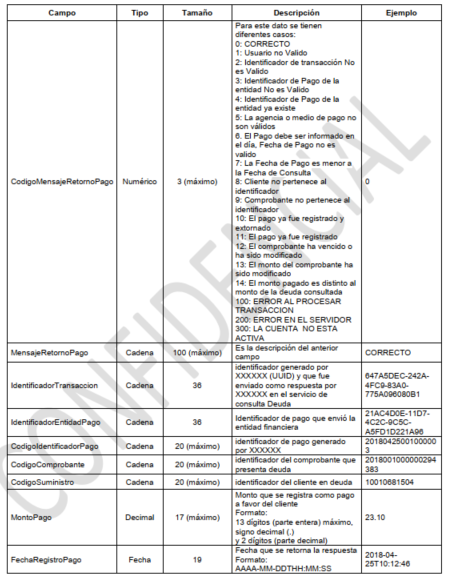
Esta operación permite registrar el pago efectuado por el cliente, para ello la entidad financiera enviará datos relacionados a la operación anterior (ConsultaDeuda), para luego obtener una respuesta la cual puede ser correcta (CodigoMensajeRetornoPago = 0, significando que se registró el pago) o incorrecta en caso haya ocurrido un error.

Al igual que en la Consulta de Deuda, en caso haya ocurrido un error, el mensaje de respuesta tendrá la descripción del error, y los demás campos sin valor.

Datos de petición



Datos de respuesta



# Controladores

# AutenticacionController

|  |  |
| --- | --- |
| **Tipo** | Controlador |
| **Descripción** | Este controlador lo usamos para todo lo es autenticación. |

Validación ValidateAntiForgeryToken .- El soporte anti-falsificación de MVC escribe un valor único en una cookie solo de HTTP y luego se escribe el mismo valor en el formulario. Cuando se envía la página, se genera un error si el valor de la cookie no coincide con el valor del formulario.

Acciones

|  |  |  |
| --- | --- | --- |
| Acción | Método | Descripción |
| Logout | GET | Limpiamos a sesión y redireccionamos al inicio |
| Login | GET | Carga Vista |
| Login | POST | Esta acción ejecuta el método post del Login, recibe 3 parámetros  email, password, CaptchaCode, hacemos las siguientes validaciones:   * Ingreso de email, password, CaptchaCode * CaptchaCode valiamos con la sesion almacenada cunado se generó la imagen * Validamos email con UsrcorreoPrimario y password con Usrcontrasena con la base de datos * Si lo encontramos en la base de datos y su correo esta validado procedemos almacenas las variables sesión |
| GetCaptchaImage | GET | Es una acción devuelve una imagen png  Creamos un código captcha y lo almacenamos en una sesión CaptchaCode,  con este código generamos una imagen. |
| ConfrimacioDeCorreo | GET | Confirma el correo de un usuario  Cuando la persona se registra se le envía un correo que tiene que confirmar,  en ese correo hay un link el cual direcciona a esta acción.  Actualiza en base de datos UsrconfirmacionCorreo a true |
| RecuperarContrasena | GET | ***Carga Vista*** |
| RecuperarContrasena | POST | Para recuperar contraseña ingresamos el email y captchaCode  Usamos el generador del captcha para crear una clave temporal y se la enviamos por correo |
| ActualizarContrasena | GET | Carga Vista |
| ActualizarContrasena | POST | Actualizamos la contraseña del usuario actual. |
| Registro | GET | Carga Vista |
| Registro | POST | Registramos a un nuevo usuario.  Al grabar en base de datos enviamos un correo para validar su correo. |
| ErrorPage | GET | Página de Error |

# PagosController

|  |  |
| --- | --- |
| **Tipo** | Controlador |
| **Descripción** | Buscamos un numero de contrato y el historial de pagos |

Acciones

|  |  |  |
| --- | --- | --- |
| Acción | Método | Descripción |
| BuscarSuministro | GET | Carga la vista |
| BuscarSuministro | POST | Buscamos un suministro, Lee los datos del web services |
| MisPagos | GET | Carga la vista |
| MisPagos | POST | Buscamos los pagos realizamos por año y mes |

# VisaController

|  |  |
| --- | --- |
| **Tipo** | Controlador |
| **Descripción** | Este controlador lo usamos para realizar el pago en Visa. |

Acciones

|  |  |  |
| --- | --- | --- |
| Acción | Método | Descripción |
| Pago | GET | Carga el Resumen de pago y en la vista está la interfaz por JavaScript con Visa |
| ConfirmacionPago | GET | Una vez realizado la acción de pagar (Formulario Visa) la respuesta de la operación si es satisfactoria o con error va a esta acción. |
| token\_seguridad | Función | Genera el Token de seguridad |
| session\_comunicacion | Función | Genera de sesión de comunicación con Visa |

# Modelos

Estas son clases generadas por Entity Framework Core que es ORM de la base de datos.

Para la generación de estas clases vamos a la “Consola de Administrador de paquetes” y ejecutamos este comando:

|  |
| --- |
| Scaffold-DbContext "Server=ELSX007;Database=dbPagosVISA;user id=UsuarioPagosVISA;password=PagosVIS@;" Microsoft.EntityFrameworkCore.SqlServer -OutputDir Models -Force |

* ElectrosurContext
* PdpOpepOperacion
* PdpPagpPago
* PdpUsrtUsuarioDelSistema

ModelViews

* ErrorViewModel

# Ayudas (Helper)

* CaptchaResult

|  |  |
| --- | --- |
| **Tipo** | Clase genérica |
| **Descripción** | Clase contenedora de los parámetros |

* Captcha

|  |  |
| --- | --- |
| **Tipo** | Clase genérica |
| **Descripción** | Genera el Captcha  Por NuGet se importa esta librería “System.Drawing.Common” |

Acciones

|  |  |
| --- | --- |
| Acción | Descripción |
| GenerateCaptchaCode | Genera las 4 letras o números aleatorios |
| ValidateCaptchaCode | Validamos el captcha con la sesión |
| GenerateCaptchaImage | Genéranos el captcha |

* PasswordHash

|  |  |
| --- | --- |
| **Tipo** | Clase genérica |
| **Descripción** | Esta clase encripta la contraseña en un hash para guardarlo en la base de datos |

Acciones

|  |  |
| --- | --- |
| Acción | Descripción |
| GetMd5Hash | Encripta la contraseña en un hash |

* SendEmailOutlook

|  |  |
| --- | --- |
| **Tipo** | Clase genérica |
| **Descripción** | Para en envió de correos |
| **Ubicación de plantillas de correo** | /wwwroot/plantilla\_email |

Acciones

|  |  |
| --- | --- |
| Acción | Descripción |
| Send | Envió de correo |

Clases de contenedoras Visa

Estas clases las usamos para serializar la respuesta de Visa

* VisaInterfaz
* ClsTokenSession
* Header
* Fulfillment
* Order
* Token
* DataMap
* RootObject
* HeaderError
* DataError
* RootObjectError

# Configuración

* AuthenticationFilter

|  |  |
| --- | --- |
| **Tipo** | IActionFilter |
| **Descripción** | Esta clase es un filtro que se ejecuta antes de ingresar al controlador, con esta clase podemos validar si el usuario tiene una sesión si no lo saca del sistema |

Acciones

|  |  |
| --- | --- |
| Acción | Descripción |
| OnActionExecuting | Esta función se encarga de validar las acciones (urls) que son anónimas o requieren autenticación.  Valida si el usuario esta logueado de tal forma que solo entra al sistema los usuarios registrados. |
| OnActionExecuted | En blanco |

* Program

|  |  |
| --- | --- |
| **Tipo** | Genérica |
| **Descripción** | Clase propia de la aplicación sin cambio alguno |

* Startup

|  |  |
| --- | --- |
| **Tipo** | Genérica |
| **Descripción** | Esta clase es la que inicia la aplicación y en donde configuramos los servicios |

Acciones

|  |  |
| --- | --- |
| Acción | Descripción |
| ConfigureServices | Configuramos lo siguiente   * Sesión * Dase de datos y cadena de conexión * Tipo aplicación en este caso es MVC que sería AddControllersWithViews * Configuraciones genéricas como AddSingleton , AddTransient |
| Configure | Activa las configuraciones anteriores como:   * Página de error * Uso de archivos estáticos * Uso de enrutamiento * Uso de autorizaciones * Uso de sesiones * Mapa de rutas |