INDICE PROYECTO FINAL

## Conexión con el CRM

1. Recursos Trailhead: <https://trailhead.salesforce.com/es/content/learn/modules/marketing-cloud-connect>
2. **AMPScript** para crear/actualizar Leads en SFMC y pasarlos al CRM:
   1. Función **RetrieveSalesforceObjects()** -> Función de control de DUPLICADOS POR EL EMAIL -> Si ya existe se Actualiza y si NO EXISTE se crea
   2. Función **UpdateSingleSalesforceObject()** -> Función para ACTUALIZAR un registro en Salesforce (SE PONE UN IF DE CONTROL
   3. Función **CreateSalesforceObject()** -> Función para CREAR un registro en Salesforce (SE PONE UN IF DE CONTROL

## CloudPage Inicio Formulario

1. Código **JavaScript:**
   1. Función **updateAdults()** -> Se activa cuando se cambian el número de adultos en el formulario. Según el número seleccionado, crea campos de entrada para ingresar el DNI
2. Código **AMPScript**:
   1. Función **Now ()** para obtener la fecha del campo “BookingDate” cuando el usuario envíe el formulario. Donde le ponemos input tipo **hidden** y **value***=”valor recogido de la función Now()”*
   2. Función **Substring(GUID(), 1, 10) ->** Se crean identificadores únicos con la función GUID() y se toman los 10 primeros caracteres utilizando Substring

## ThankYouPage\_API\_SSJS

1. **Recurso llamada API con SSJS** obtenido en: [*https://ampscript.xyz/how-tos/how-to-trigger-journey-api-entry-event-ampscript-ssjs/*](https://ampscript.xyz/how-tos/how-to-trigger-journey-api-entry-event-ampscript-ssjs/)
2. **AMPScript**:
   1. Recogemos variables con el método **RequestParameter()** y le asignamos los valores con set @nombreVariable
3. **SSJS:**
   1. Utilizamos la función **Platform.Load()** para cargar la biblioteca necesaria.
   2. Recogemos los datos de las variables en el **objeto data {}.** Hacemos un GetValue para ello.
   3. Definición de la configuración: Se define un **objeto setup{}** con configuraciones como Base URIs, ID Client y Client Secret para obtener el **token** y hacer la **llamada API**.
   4. Con **triggerEvent()** envía una solicitud HTTP para desencadenar un evento en Marketing Cloud
   5. Bloque **try-catch**: Envuelve las funciones en un bloque try-catch para manejar errores.
   6. Utiliza la función **Write()** para mostrar un mensaje en la página según si la operación fue exitosa o no.

## Email Welcome

1. Recuperación de Copies basada en el idioma del usuario:
   1. **AttributeValue**: Obtiene el valor de los atributos "Language" y "HotelName" del usuario.
   2. **LookupRows**: Busca múltiples filas en la tabla "DE\_Copies" en base al idioma del usuario.
   3. **rowcount**: Cuenta el número de filas devueltas.
   4. **row y field**: Si hay filas, recoge los distintos campos como "Greeting", "Main", etc.
2. Recuperación de información del Hotel
   1. **LookupRows**: Busca múltiples filas en la tabla "DE\_Hotels" en base al nombre del hotel.
   2. **rowcount, row y field**: Si hay filas, recoge los campos como "Country", "City", etc.
3. Validaciones y obtención de Moneda
   1. **EMPTY y RaiseError**: Realiza una validación para asegurarse de que el idioma está presente.
   2. **LookupRows**: Busca en "DE\_Currency" para encontrar el símbolo de moneda y la tasa de conversión.
4. Validación del hombre del Hotel y precio por noche
   1. **EMPTY y RaiseError**: Realiza una validación para el nombre del hotel.
   2. **LookupRows**: Busca en "DE\_Hotels" para encontrar el precio por noche del hotel.
5. Cálculos
   1. **Lookup, Multiply, y empty**: Realiza cálculos para el costo total y el costo total convertido.
6. Recuperación de textos para precio y moneda convertida
   1. **LookupRows, rowcount, row, y field**: Busca en "DE\_Copies" para encontrar los textos relacionados con la moneda y el precio.
7. Botones en el correo
   1. **LookupRows, rowcount, row, y field:** Busca en "DE\_Copies" para encontrar los textos de los botones como "Cancelar reserva" o "Confirmar".

## CP\_ConfirmedBooking

1. Se recogen los datos pasados por el botón con **RequestParameter("Booking\_Id")**’
2. Se **actualiza el Estado de la Reserva** con la **función UpdateDE():**
   1. *set @result = UpdateDE("DE\_Bookings", 1, "Booking\_Id", @Booking\_Id, "Status", "****Confirmed****")* 🡪 Campo **Status**
3. Se utiliza la **función Lookup** para Consultar el **valor de Status Actualizado**
   1. *set @statusValue = Lookup("DE\_Bookings", "Status", "Booking\_Id", @Booking\_Id)*
4. Y luego **se pinta el CAMPO STATUS con** **%%=v(@statusValue)=%%** para ver que se ha actualizado bien, y ahora tenga valor **“*Confirmed*”**

## CP\_CanceledBooking

1. Se recogen los datos pasados por el botón por **RequestParameter("Booking\_Id")**’
2. Se **actualiza el Estado de la Reserva** con la **función UpdateDE():**
   1. *set @result = UpdateDE("DE\_Bookings", 1, "Booking\_Id", @Booking\_Id, "Status", "****Canceled****")* 🡪 Campo **Status**
3. Se utiliza la **función Lookup** para Consultar el **valor de Status Actualizado**
   1. *set @statusValue = Lookup("DE\_Bookings", "Status", "Booking\_Id", @Booking\_Id)*
4. Y luego **se pinta el CAMPO STATUS con** **%%=v(@statusValue)=%%** para ver que se ha actualizado bien, y ahora tenga valor **“*Canceled*”**

## CP\_ModifyBooking1

1. AMPScript:
   1. Se recuperan todos los parámetros de la reserva con **“Requestparameter()”**
   2. Se pintan en pantalla todos los parámetros de la reserva con **%%=v(@nombreVaviable)=%%**
   3. Se pinta el formulario con los diferentes campos
   4. Se utiliza la función **AttributeValue(“NameInput”)** para recoger los valores que el usuario ha puesto en el formulario para modificar
   5. En el **ACTION** del form se envían todos los datos de las variables a la **CP\_ModifyBooking2**, donde se hará un update de estoso datos en la reserva.

## CP\_ModifyBooking2

1. Se recogen todas las variables que se pasan del formulario de “CP\_ModifyBooking1” con **RequestParameter()**
2. Se Actualiza la Data Extension con la función **UpdateData()** 🡪
   1. *set @rowsUpdated = UpdateData("DE\_Bookings", 2, "Booking\_Id", @Booking\_Id, "Contact\_ID", @Contact\_ID, "HotelName", @HotelName, "Type", @Type, "Adults", @Adults, "Kids", @Kids, "Rooms", @Rooms, "Checkin", @Checkin, "Checkout", @Checkout, "Contact", @Contact, "EmailAddress", @EmailAddress, "Phone", @Phone, "Nights", @Nights, "Status", @Status, "DateBooking", @DateBooking, "Language", @Language)]%%*
3. Se pintan todos los datos actualizados con **%%=v(@valorVariableCampo)=%%**

## CP\_Unsuscribed

1. **AMPScript:**
   1. Recogemos parámetros con **RequestParameter()** para recoger valores del Contact\_ID y el EmailAddress enviados por el enlace del Email
   2. Pintamos por pantalla con **%%=v(@landingContactID)=%%** las datos recogidos en variables del Usuario que se ha dado de Baja de nuestras comunicaciones
   3. Se verifican si las **variables** están **vacías** o no antes de proceder con un **IF NOT EMPTY (variable)… ENDIF**
   4. Insertamos en la Data Extension DE\_Unsuscribed con la función InsertDaata (“DE\_Unsuscribed”, “nombreCampo1”, @valorCampo1, “nombreCampo2”, @valorCampo2)
   5. Se utiliza otro IF – ELSE – ENDIF para informar al usuario si los datos se han establecido correctamente o ha habido un error

## Otro titulo

1. asdfasdf