

Diseño de una aplicación Angular

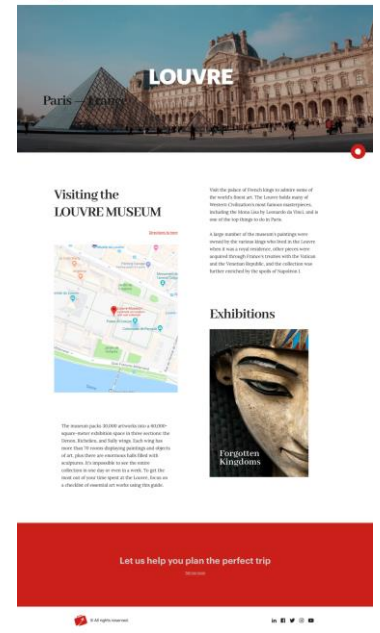
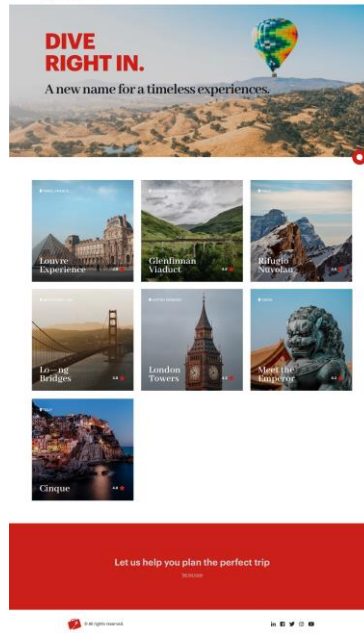
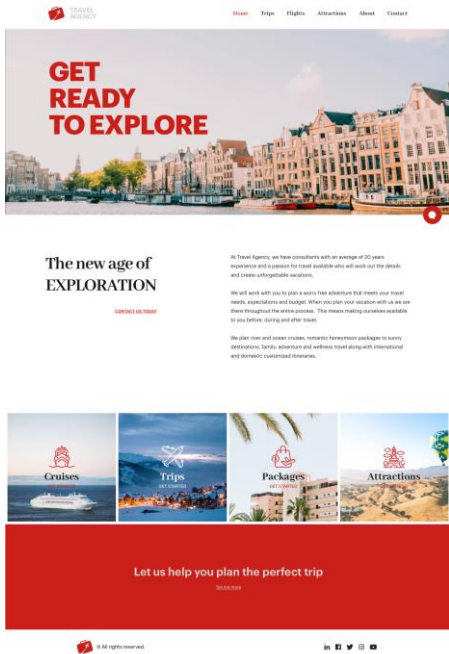
Importación de un diseño desde Sketch

GeneXus™

En otro video vimos cómo realizar nosotros mismos las definiciones de diseño de nuestra aplicación utilizando el Design System Object.

En este video veremos cómo podemos importar el diseño de pantallas completas, son especificación de colores, tipos de letras y demás, hecho por un profesional en diseño de aplicaciones digitales, en Sketch, su herramienta habitual de trabajo, e importar ese diseño en nuestra aplicación GeneXus.

Diseño presentado por el diseñador



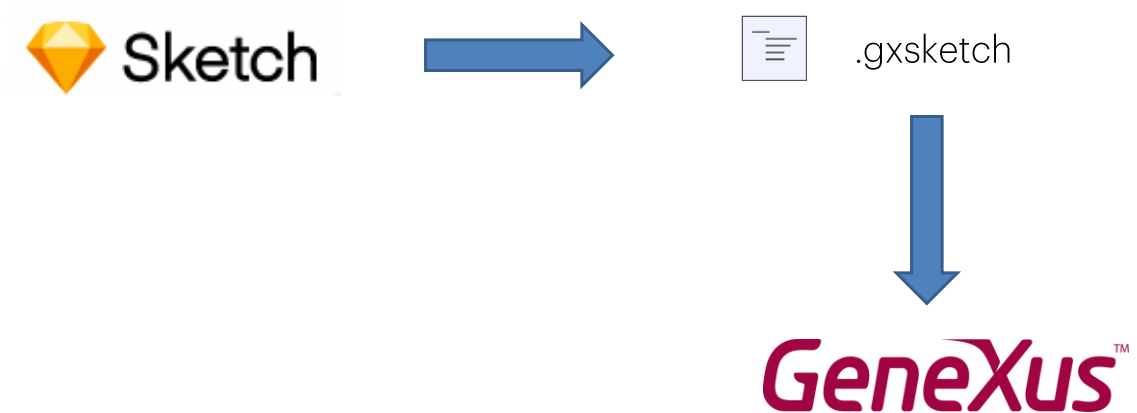
Hemos acordado con el diseñador que necesitamos una pantalla inicial que nos sirva de punto de partida para el resto de las pantallas de la aplicación, otra pantalla para ver la lista de atracciones turísticas disponibles y una tercera pantalla de detalle de una atracción turística.

Estas son las pantallas confeccionadas por el diseñador. En la de la izquierda que es la pantalla principal, vemos arriba del todo el logo de la aplicación y un menú de acciones. Más abajo, una imagen de fondo que tiene sobre ella un título destacado y abajo a su derecha un botón para iniciar un chatbot, luego una sección con datos de la agencia de viajes, debajo de ésta cuatro botones para acceder a los datos de las entidades que nuestra aplicación va a manejar y debajo de éstos un footer con un marco rojo con un mensaje y los links a las redes sociales.

Luego la pantalla del medio muestra en la parte superior el logo, el menú, una imagen contextual con título y botón de chatbot, una grilla de las atracciones turísticas disponibles y el footer. Vemos que muchos de estos elementos también se repiten en la tercera pantalla donde vemos el detalle de una atracción, en este caso del Louvre.

Evidentemente el diseñador ha hecho un buen trabajo, planteando un aspecto uniforme entre las pantallas, con una forma clara y limpia de presentar la información.

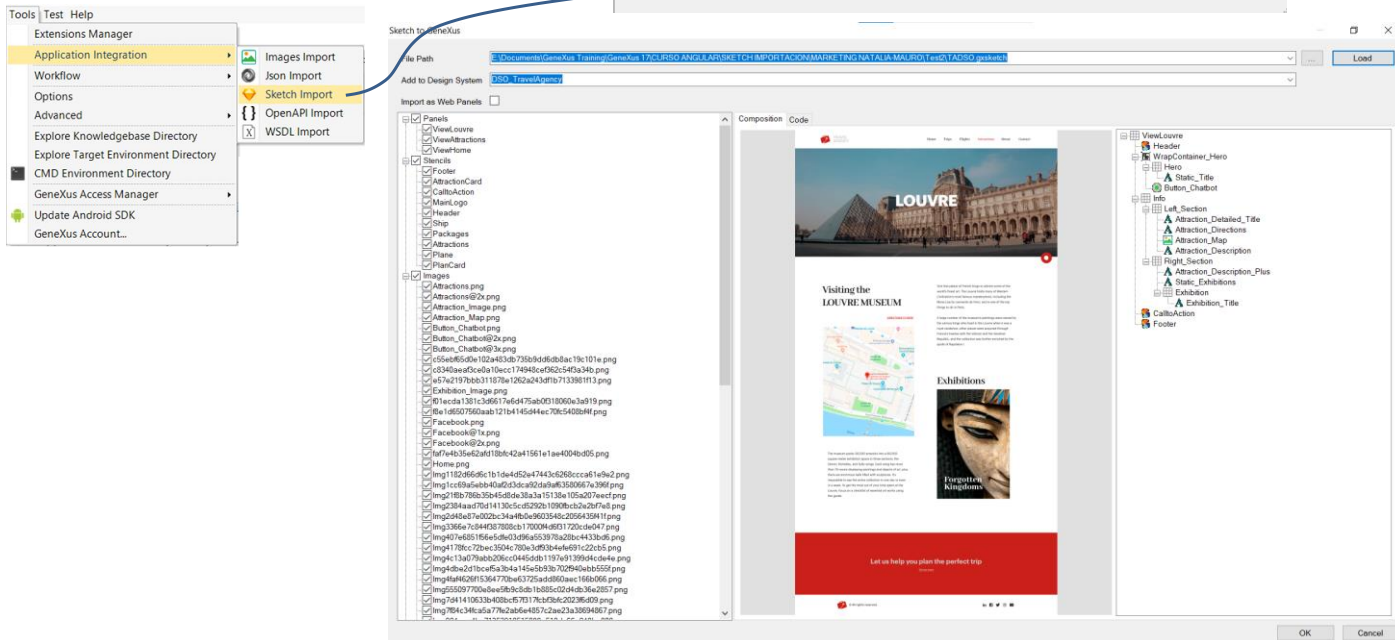
Importación desde Sketch



Este diseño lo realizó en la herramienta que usa siempre, en este caso Sketch. Luego utilizará un plugin de GeneXus instalado en Sketch que le permite crear un archivo de extensión gxsketch, que es el que nos envía.

GeneXus nos permite importar este archivo creado en la herramienta del diseñador, directamente a nuestra KB y todos los elementos de diseño quedarán automáticamente creados en un Design System Object e incluso se crearán los objetos GeneXus correspondientes con todos los controles necesarios para que funcione la aplicación, con las pantallas que vimos antes.

Importación del archivo .gxs sketch



Para importar el archivo .sketch que nos envió el diseñador, vamos a Tools/Application Integration y elegimos Sketch import.

Agregamos el nombre del Design System Object donde queremos que queden guardadas todas las definiciones de diseño.

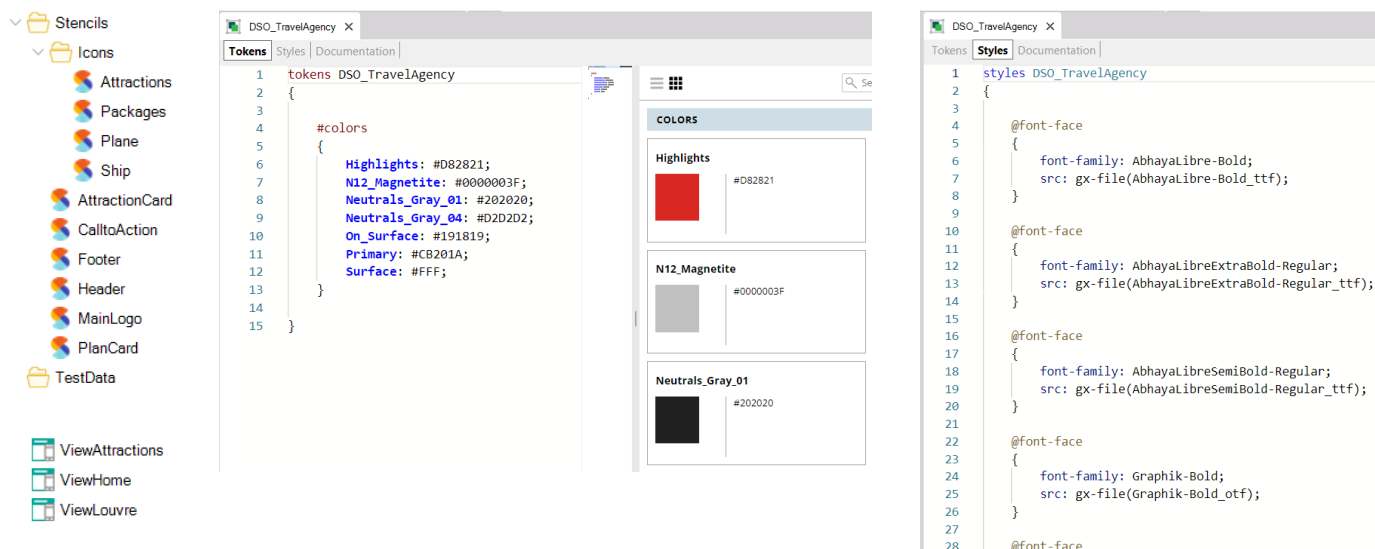
El check box Import as Web Panels lo dejamos desmarcado, porque queremos que se creen panels y no webpanels.

Si hacemos clic en alguno de los panels que la importación a crear, vemos su vista previa y corroboramos que corresponden al diseño que validamos. A la derecha se nos informa los controles que contendrá.

Vemos que también importará las imágenes para poder ejecutar el panel con datos fijos que nos permitirán ejecutar y verificar que la funcionalidad es la deseada. Luego deberemos sustituir estos datos fijos, por los datos almacenados en la base de datos de nuestra aplicación.

Estamos de acuerdo, así que presionamos OK.

Objetos creados en la importación



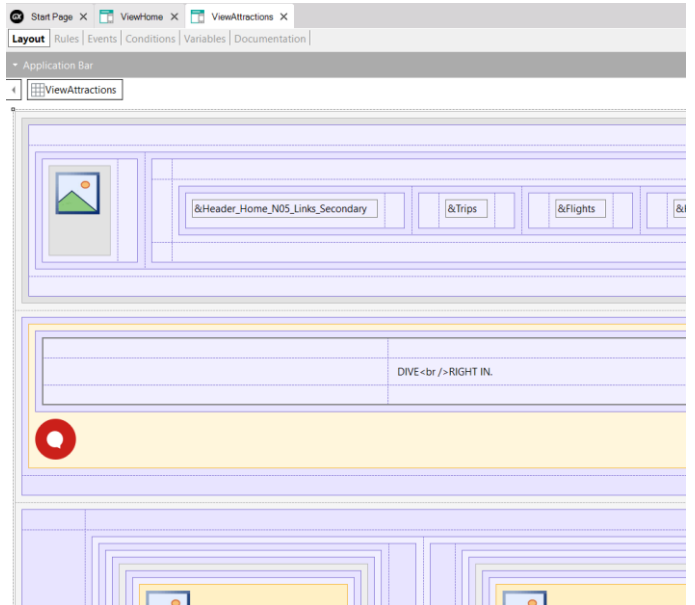
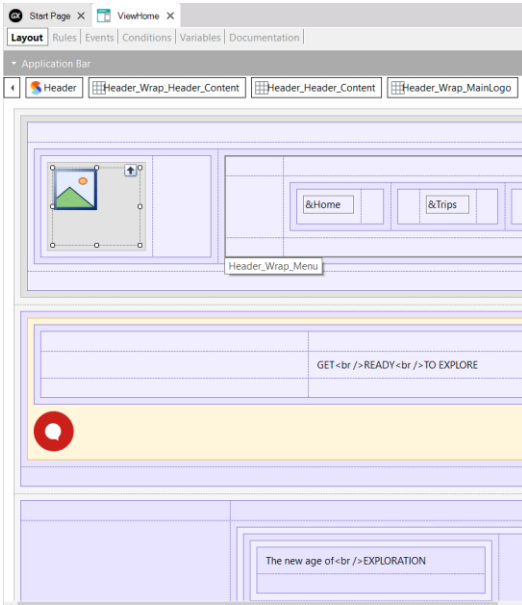
En la ventana de Output se muestra el progreso de la importación y el resultado sin errores.

En el KB Explorer, vemos que se crearon dos folders: Stencils que contiene varios stencils utilizados para encapsular diseño en la aplicación y TestData, que contiene data providers y sdt para cargar datos fijos, que nos permita probar la aplicación para ver si el diseño quedó como esperábamos.

Mas abajo vemos que se creó el Design System Object DSO_TravelAgency, que si lo abrimos vemos los Tokens y Styles que definen la apariencia de los controles en pantalla.

Abajo del todo encontramos que se crearon automáticamente los objetos panels que habíamos visto en la pantalla del wizard de importación, como el de la página inicial (ViewHome), el que muestra las atracciones (ViewAttractions) y el detalle de una atracción (ViewLouvre).

Objetos panels creados automáticamente en la importación



Si vemos el form del ViewHome observamos que ya están incluidos todos los componentes para el contenido visual, como , tablas, variables, imágenes, etc. todo creado automáticamente en el proceso de importación. Y lo mismo para el panel ViewAttractions.

Objetos creados en la importación

The screenshot displays the GeneXus IDE interface. On the left, a small preview window shows a button labeled '&Logo' with a green triangle icon. The main window is titled 'ViewHome X' and has the 'Variables' tab selected. It shows a list of variables with columns for Name, Type, Is Collection, and Description. The 'Logo' variable is highlighted in blue. To the right, the 'Properties' panel is open for the '&Logo' variable, showing its Name, Description, Column title, and Class (Image). The 'Type Definition' section shows it is based on '(none)' and has a data type of 'Image'. The 'Initial value' is set to 'Image:Logo.link()'. A blue arrow points from the '&Logo' variable in the list to the '&Logo' button in the preview window.

Name	Type	Is Collection	Description
Contact	VarChar(40)	<input type="checkbox"/>	Contact
Cruises	VarChar(40)	<input type="checkbox"/>	Cruises
Facebook	Image	<input type="checkbox"/>	Facebook
Flights	VarChar(40)	<input type="checkbox"/>	Flights
Home	VarChar(40)	<input type="checkbox"/>	Home
Information_Text	LongVarChar(2M)	<input type="checkbox"/>	Information_Text
Instagram	Image	<input type="checkbox"/>	Instagram
ISO1	Image	<input type="checkbox"/>	ISO1
Let_us_help_you_plan	VarChar(40)	<input type="checkbox"/>	Let_us_help_you_plan
link_information	VarChar(40)	<input type="checkbox"/>	link_information
LinkedIn	Image	<input type="checkbox"/>	LinkedIn
Logo	Image	<input checked="" type="checkbox"/>	Logo
Packages	Image	<input type="checkbox"/>	Packages
Plan_Image	Image	<input type="checkbox"/>	Plan_Image
PlanCardAttractions_Cr...	VarChar(40)	<input type="checkbox"/>	Plan Card Attractions_Cruises
PlanCardAttractions_Fla...	Image	<input type="checkbox"/>	Plan Card Attractions_Plan_Image
PlanCardCruises1 Cruis...	VarChar(40)	<input type="checkbox"/>	Plan Card Cruises1 Cruises N01 Titles H4 Negative

Variable: &Logo	
Name	Logo
Description	Logo
Column title	Logo
Class	Image
Exo Gen Definition Ve	
Type Definition	
Based on	(none)
Data Type	Image
Collection	False
Initial value	Image:Logo.link()

Si abrimos el panel ViewHome, y vamos a ver las variables, vemos que las imágenes se cargaron mediante la propiedad Initial Value, haciendo referencia a las imágenes que el proceso de importación cargó en la KB.

Y si vamos a los eventos, vemos que también está programada la interacción entre paneles, ya que al presionar el botón de atracciones, nos invoca al panel ViewAttractions.

Objeto de startup de nuestra aplicación

The screenshot shows the GeneXus IDE interface. On the left, a tree view shows the project structure with 'Menu' expanded to show 'Items' containing 'Action (ViewAttractions)', 'Action (ViewHome)', and 'Action (ViewLouvre)'. The main editor window is titled 'Menu' and has the 'Events' tab selected. It contains the following code:

```

1  /* Generated by GeneXus Design Import [Start] */
2
3  Event 'ViewAttractions'
4    ViewAttractions()
5  EndEvent
6
7  Event 'ViewHome'
8    ViewHome()
9  EndEvent
10
11 Event 'ViewLouvre'
12   ViewLouvre()
13 EndEvent
14
15 /* Generated by GeneXus Design Import [End] */
16

```

On the right, the Properties panel is open for 'Menu: TADSOMenu'. The properties are as follows:

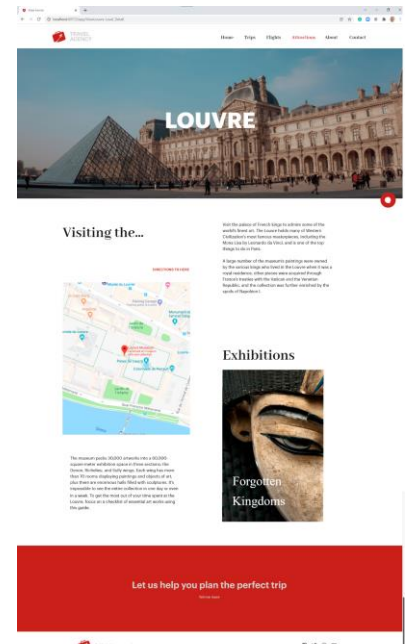
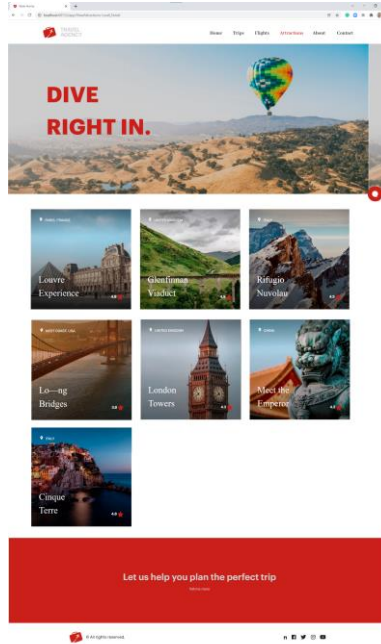
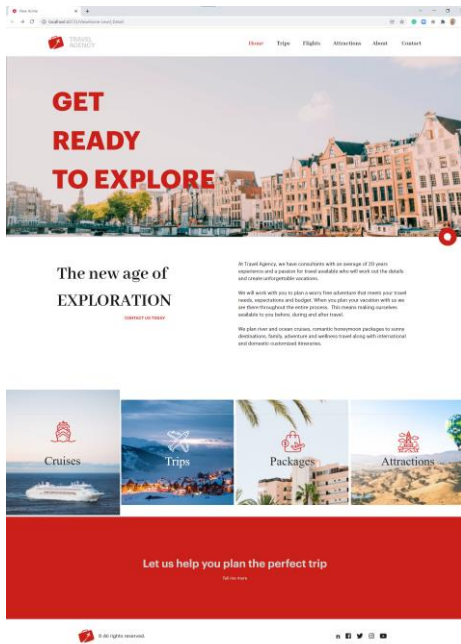
Name	TADSOMenu
Description	TADSOMenu
Module/Folder	Root Module
Qualified Name	TADSOMenu
Object Visibility	Public
Automatic Update	False
Main program	True
Master Panel	(none)
Generate OpenAPI into Use Environment	
Miscellaneous	
Generate Object	True
Network	
Connectivity Support	Online

Aquí podemos ver que se también creó un objeto menú, que por defecto sería el objeto main a ejecutar. Vemos que en los eventos, se invocan a los otros paneles de la aplicación.

Este objeto se verá solamente en la generación para plataforma nativa, en el caso de Angular, el objeto que se va a ver primero es el ViewHome, que es la segunda opción del menú.

Para ejecutar, ponemos al objeto ViewHome como Startup Object y damos F5.

La aplicación en ejecución con datos fijos



Vemos que se ejecuta la pantalla inicial y que quedó tal cual lo previó nuestro diseñador!.

Hacemos clic en Attractions y vemos la lista de atracciones turísticas de la agencia de viajes.

Recordemos que estamos viendo los datos fijos que se cargaron durante la importación.

Si hacemos clic en el Louvre, vemos que nos aparece un página con información detallada, todo con un diseño prolijo y estéticamente agradable.

Este ejemplo nos permitió ver cómo es el proceso de desarrollo de una aplicación y lo importante que es trabajar en equipo con personas con diferentes perfiles, cada uno aportando a la mejor solución.

Modificación de los nuevos objetos para que accedan a datos de la base de datos

```

Layout | Rules | Events | Conditions | Variables | Documentation
Events
1  /* Generated by GeneXus Design Import [Start] */
2
3  Event Grid_Attractions.Load
4      Composite
5          For &ViewAttractions_Grid_Attractions_SDT in ViewAttractions_Grid_Attractions_DP()
6              Grid_Attractions.ItemLayout = &ViewAttractions_Grid_Attractions_SDT.Layout
7              load
8          EndFor
9      EndComposite
10 EndEvent
  
```

Name	Type
ViewAttractions_Grid_Attractions_SDT	
Layout	VarChar(40)
Attraction_Image	Image
Opacity_Adjustment_2	Image
Attraction_Location	VarChar(40)
Location_Icon	Image
Opacity_Adjustment_1	Image
Attraction_Name	VarChar(40)
Star_Icon	Image
Rating_Value	VarChar(40)

```

Start Page X | ViewAttractions_Grid_Attractions_DP X
Source | Rules | Variables | Help | Documentation
1  ViewAttractions_Grid_Attractions_DP
2  {
3
4      ViewAttractions_Grid_Attractions_SDT
5      {
6          Layout = "AttractionItem4"
7          Attraction_Image = Image:e04d854f986edeff804aafef50f757e5120c94ce.Link()
8          Opacity_Adjustment_2 = Image:Opacity_Adjustment_2.Link()
9          Attraction_Location = Upper("Italy")
10         Location_Icon = Image:Location_Icon.Link()
11         Opacity_Adjustment_1 = Image:Opacity_Adjustment_1.Link()
12         Attraction_Name = "Rifugio" + NewLine() + "Nuvo!au"
13         Star_Icon = Image:Star_Icon.Link()
14         Rating_Value = "4.5"
15     }
16 }
17 ViewAttractions_Grid_Attractions_SDT
18 {
19     Layout = "AttractionItem4"
20     Attraction_Image = Image:b8d1a9828c1a76bdbd62713259d82fc5055db4c1.Link()
21     Opacity_Adjustment_2 = Image:Opacity_Adjustment_2.Link()
22     Attraction_Location = Upper("United Kingdom")
23     Location_Icon = Image:Location_Icon.Link()
24     Opacity_Adjustment_1 = Image:Opacity_Adjustment_1.Link()
25     Attraction_Name = "Glenfinnan" + NewLine() + "Viaduct"
26     Star_Icon = Image:Star_Icon.Link()
27     Rating_Value = "4.9"
28 }
29 ViewAttractions_Grid_Attractions_SDT
  
```

Hasta ahora estuvimos viendo datos fijos de prueba, que los diseñadores agregaron para que podamos ver el comportamiento de la aplicación con el nuevo diseño. Así que vamos a hacer algunos cambios, para que podamos ver los datos reales de nuestra base de datos.

Si vamos a Los eventos del panel ViewAttractions, vemos que la grilla se carga utilizando una variable tipo SDT de nombre &ViewAttractions_Grid_Attractions_SDT y el data provider ViewAttractions_Grid_Attractions_DP.

Vamos a sustituir la carga del DP para que en lugar de tomar datos fijo, utilice los atributos de la tabla Attraction y de la tabla Country.

Modificación de los nuevos objetos para que accedan a datos de la base de datos

```

1 ViewAttractions_Grid_Attractions_DP_BD from Attraction
2 {
3   ViewAttractions_Grid_Attractions_SDT
4   {
5     Layout = "AttractionItem"
6     Attraction_Image = AttractionPhoto
7     Opacity_Adjustment_2 = Image:Opacity_Adjustment_2.Link()
8     Attraction_Location = CountryName
9     Opacity_Adjustment_1 = Image:Opacity_Adjustment_1.Link()
10    Attraction_Name = AttractionName
11    Star_Icon = Image:Star_Icon.Link()
12    Rating_Value = str(AttractionRating)
13  }
14 }
15

```

```

1 /* Generated by GeneXus Sketch Import [Start] */
2
3 Event Grid_Attractions_Small.Load
4   For &ViewAttractions_Grid_Attractions_Small_SDT in ViewAttractions_Grid_Attractions_Small_DP_BD()
5     Grid_Attractions_Small.ItemLayout = &ViewAttractions_Grid_Attractions_Small_SDT.Layout
6     load
7   EndFor
8 EndEvent

```

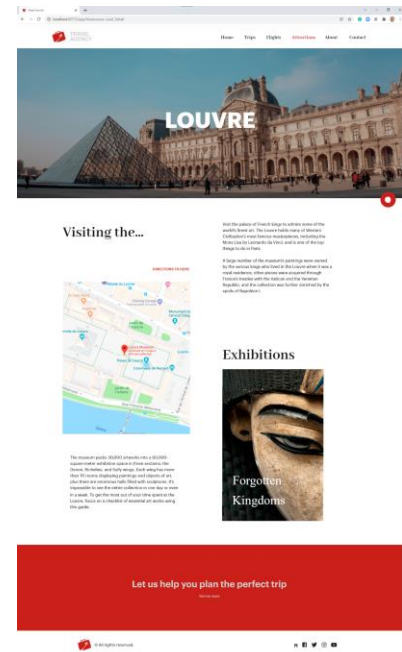
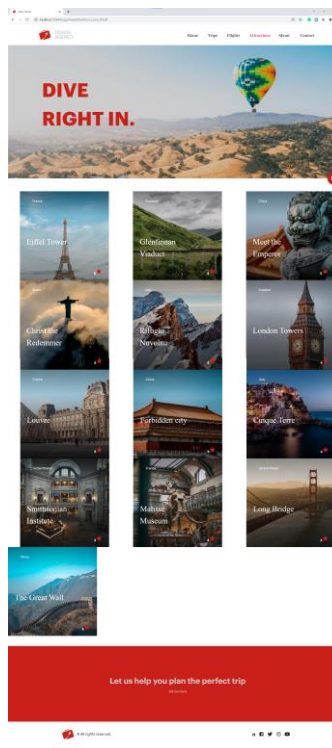
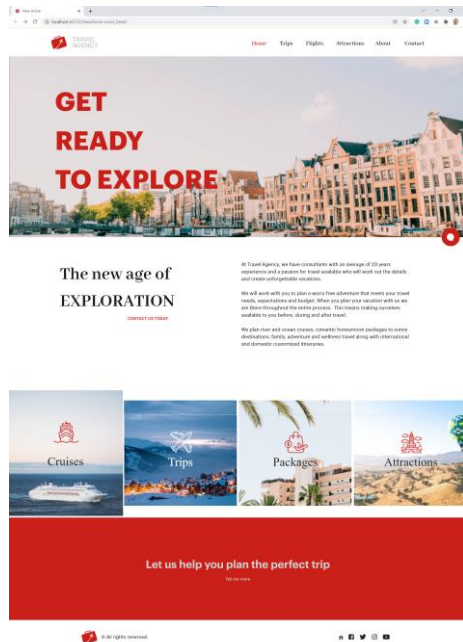
Al Data Provider le hacemos Save As y guardamos el nuevo data provider con el nombre ViewAttractions_Grid_Attractions_DP_BD.

Agregamos la cláusula from Attractions, y cargamos el campo Attraction_image con el valor del atributo AttractionPhoto, el campo Attraction_name con el atributo AttractionName, a Attraction_location le asignamos el atributo CountryName y a Attraction_rating le asignamos el valor del atributo AttractionRating. Borramos todo lo que no nos sirve.

Ahora volvemos al panel ViewAttractions y cambiamos el data provider que carga las atracciones, por el nuevo que construimos.

Salvamos, damos botón derecho sobre el panel ViewHome y elegimos Run.

Ejecución de la aplicación con datos reales



Si vamos a las atracciones, vemos que estamos recorriendo las atracciones que teníamos cargadas en la base de datos, ya que vemos a la Torre Eiffel, o al Viaducto Glenfinn que en los datos fijos no estaban y si paginamos vemos que están todas las atracciones. Si hacemos clic en Louvre vemos la página del detalle de la atracción.

Estamos cumpliendo nuestro objetivo pedido por la Agencia de mostrar todas las atracciones turísticas, pero ahora con la aplicación con un diseño más apropiado.

No cabe duda de que si disponemos de un diseñador en nuestro equipo, nunca haríamos el trabajo de escribir a mano el DSO desde cero, sino que la importación de estas definiciones desde Sketch nos resuelve el diseño en forma más profesional y podemos dedicarnos a la parte funcional de nuestra aplicación.

Esta forma de trabajar en la que integramos el diseño de un profesional a nuestro proyecto GeneXus nos brinda muchas ventajas, ya que se optimiza el esfuerzo, haciendo que cada integrante del equipo aporte según su perfil, en lo que mejor hace.

En siguientes videos abordaremos otros temas como agregamos seguridad a nuestra aplicación o cómo hacemos para ponerla en producción en las instalaciones del cliente.

GeneXus™

training.genexus.com
wiki.genexus.com
training.genexus.com/certifications