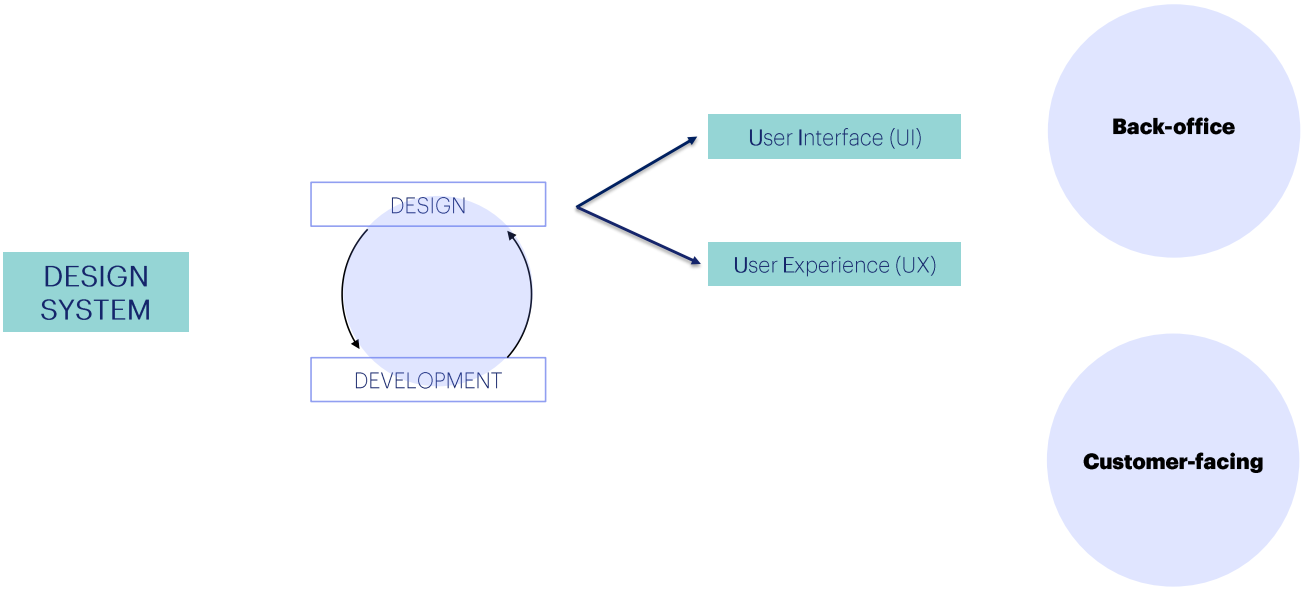
Pantallas web con foco en Customer-facing

Design System

GeneXus™

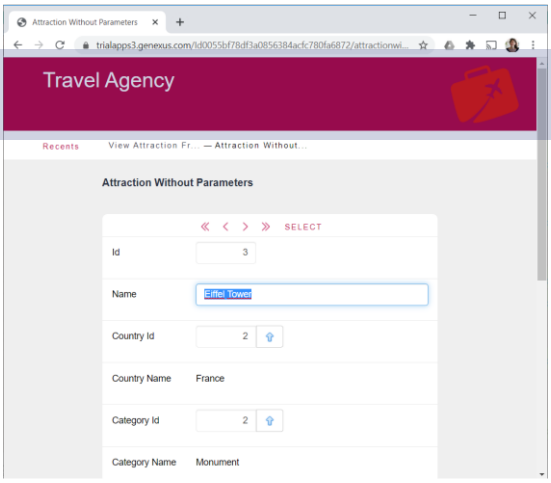
Hasta ahora no nos hemos preocupado del diseño de nuestros paneles, sino que nos hemos concentrado en su funcionamiento. Veamos ahora como incorporar diseño a nuestra aplicación.

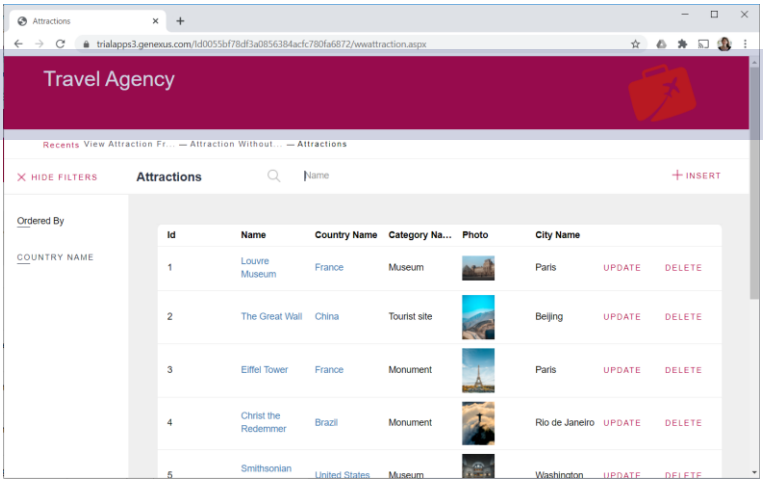


En los videos anteriores vimos que tanto las aplicaciones de Back-office como las Customer-facing tienen requerimientos de diseño y usabilidad que condujeron a la introducción del concepto de Design System.

El Design System involucra al conjunto de Principios, Patrones y Prácticas que proveen a nuestra aplicación de coherencia, uniformidad y robustez. Esto determina que el ciclo de desarrollo involucre tanto a desarrolladores como diseñadores.

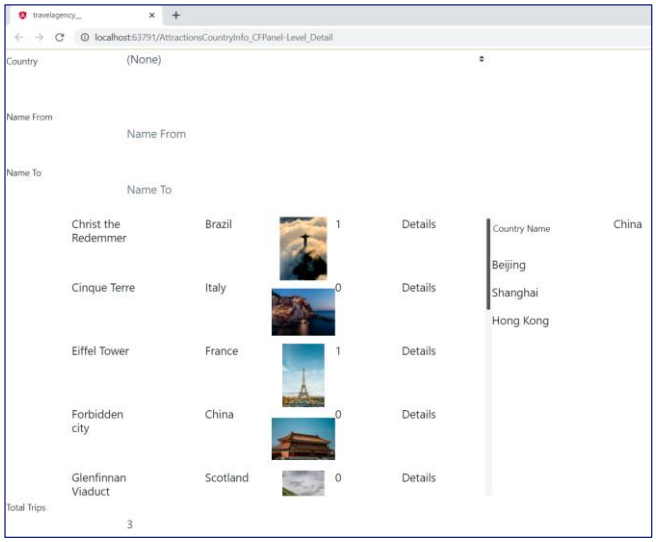
Design System predeterminado en una Web app

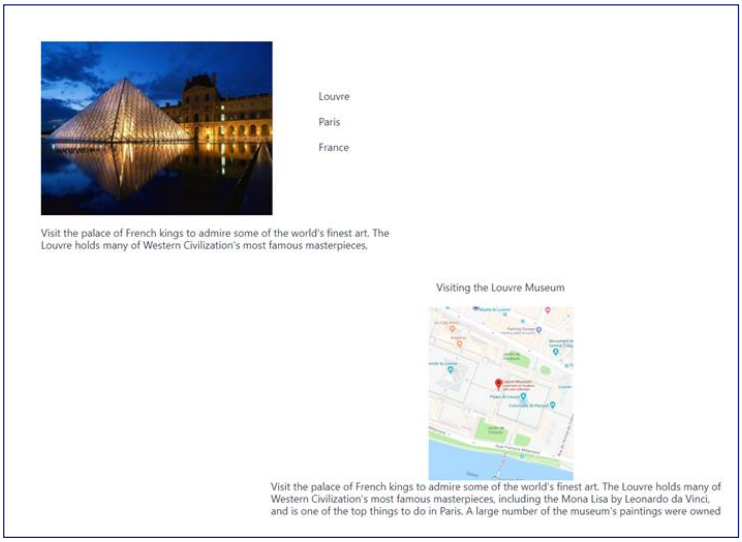




Vimos también que tanto las transacciones como los webpanels que integraban el back-office, contaban con un Design System predeterminado.

Design System predeterminado en customer-facing



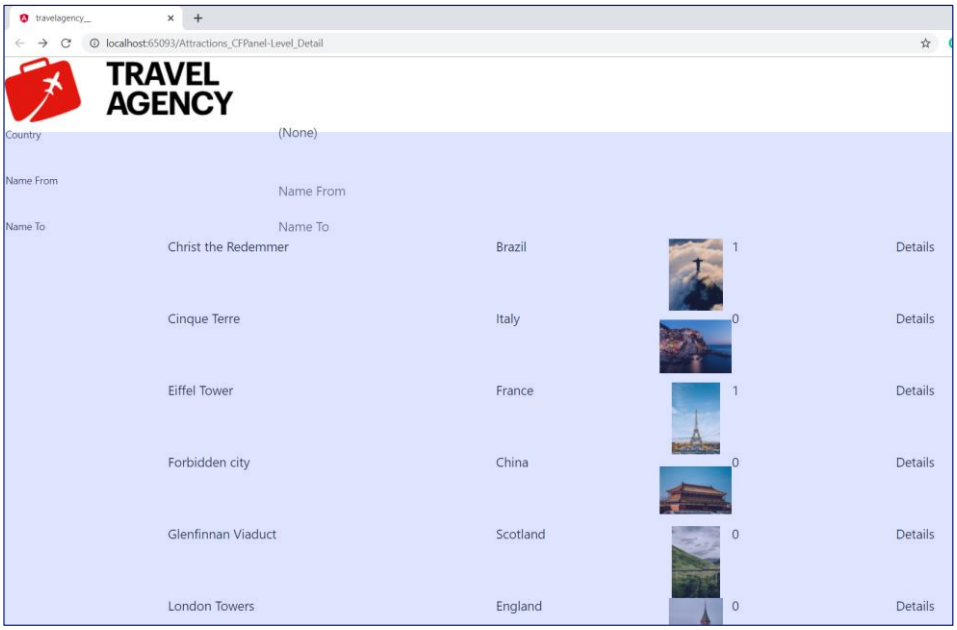


Al igual que en la aplicación de back-office, en los objetos que usamos en la parte customer-facing, también hay un diseño predefinido incorporado por GeneXus.

A diferencia de los objetos transacción y webpanels que tenían asignado por defecto una master page, en los objetos del front-end customer-facing, no hay un objeto Master Panel asignado por defecto que provea un contenedor donde ejecute la´aplicación
e incorpore componentes básicos de diseño en la ejecución.

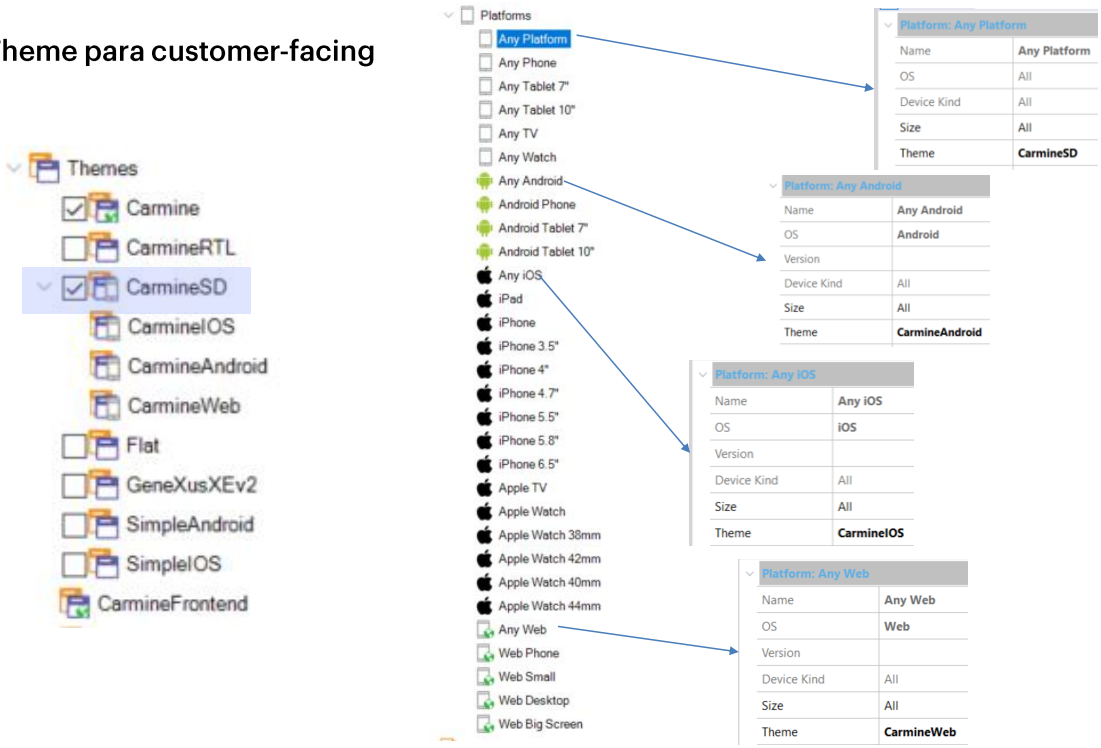
Sin embargo, podemos observar que los controles siguen un diseño y tienen una coherencia de aspecto, en forma predeterminada. También vimos que es posible asignar un objeto Master Panel a los paneles de la aplicación como vimos

Design System predeterminado en customer-facing



Sin embargo, podemos observar que los controles siguen un diseño y tienen una coherencia de aspecto, en forma predeterminada. También vimos que es posible asignar un objeto Master Panel a los paneles de la aplicación como vimos

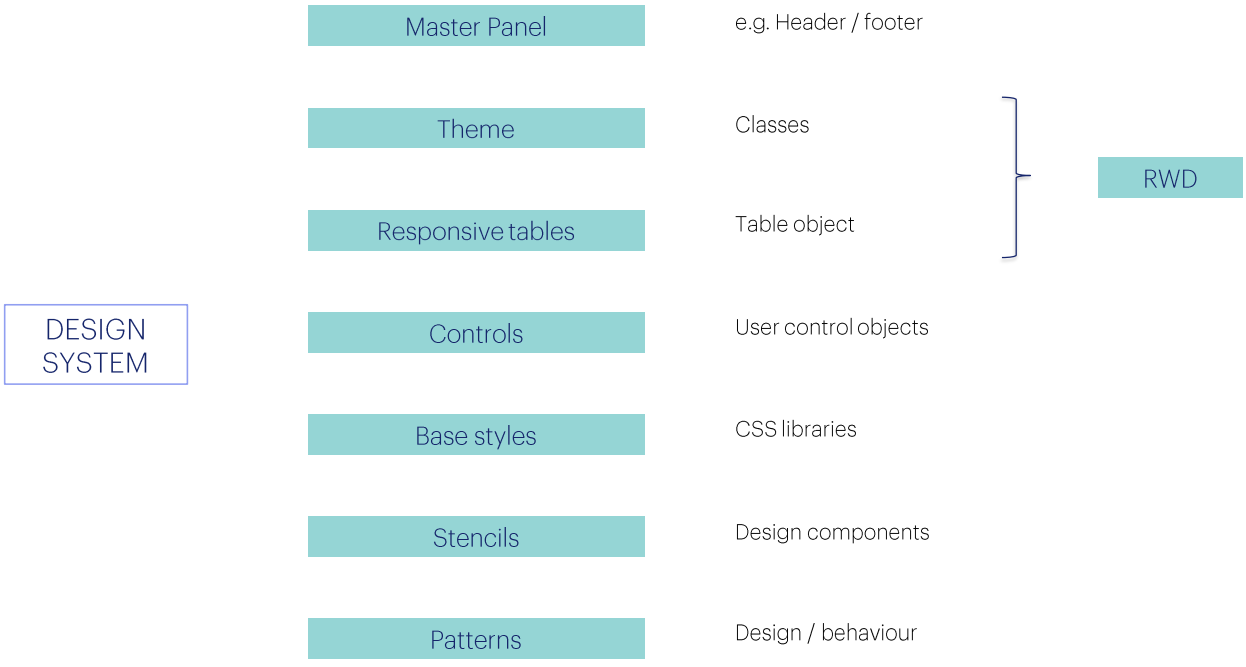
Objeto Theme para customer-facing



La apariencia de los controles en pantalla, se toman a partir de las definiciones del objeto Theme de nombre CarmineSD.

Luego dependiendo de la plataforma elegida, se utilizará uno de los subtemas, por ejemplo cuando generamos en Android se utiliza CarmineAndroid, para Apple se utiliza CarmineIOS y cuando generamos Angular, se toman las definiciones del tema CarmineWeb.

Esto lo podemos verificar si en el KB Explorer abrimos el nodo Platforms y para cada plataforma vemos su propiedad Theme. Por ejemplo para Any Platform vemos que está asignado el tema CarmineSD, para Any Android el tema CarmineAndroid, para Any iOS el tema CarmineIOS y para Any Web el tema CarmineWeb.



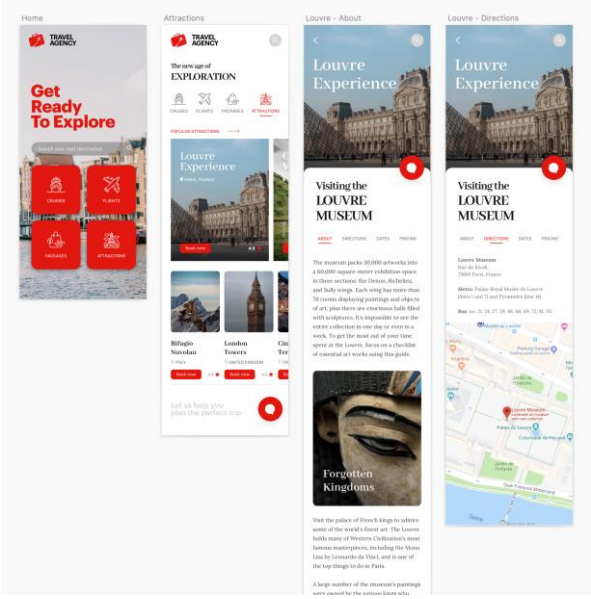
Con respecto a los componentes que vimos que integran un Design System, en una aplicación customer-facing también contamos con un master panel, objetos theme y clases. No hay un control dedicado a tablas responsivas, ya que el control Table permite la adaptación automática del contenido cuando se genera una aplicación web con Angular.

A diferencia del pattern Work With para web que genera objetos webpanels, el patrón Work With para objetos customer-facing no genera objetos panels, sino que genera un objeto específico denominado WorkWithDevices.

El resto de los componentes del Design System están representados, más allá de que la implementación de cada uno varía con la plataforma (web o nativa) y con el generador utilizado (Angular, Android o Apple).

Al igual que cuando vimos la personalización del diseño en la aplicación de back-office, en las aplicaciones customer-facing es posible trabajar con los distintos elementos del Design System (clases, temas, controles, etc.) para lograr un diseño que permita la mejor experiencia de usuario.

Importación de un diseño desde Sketch

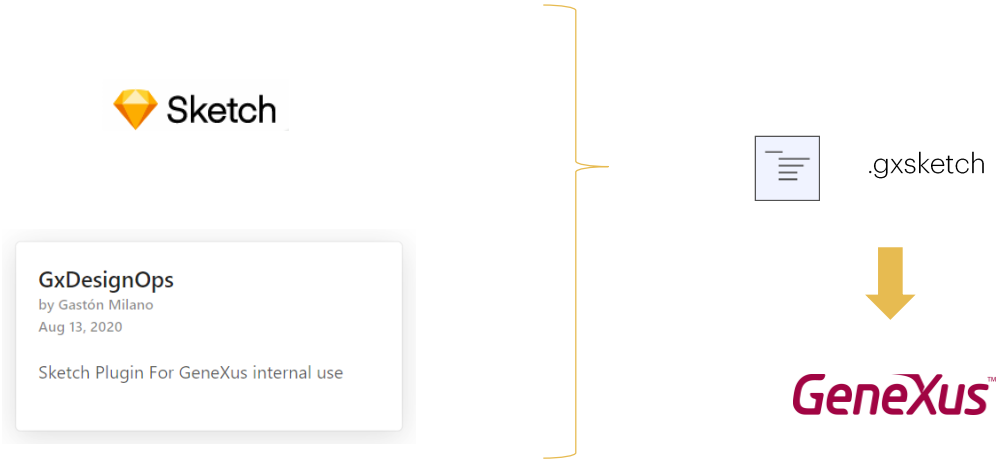


Sin embargo, el mejor camino y el más sencillo, es que un diseñador defina eso por nosotros e importemos el diseño en nuestra KB.

Queremos crear una página inicial con botones a distintas partes de la aplicación, pero nos interesa solamente ver los datos de las atracciones turísticas.
Al presionar el botón de las atracciones queremos ver una lista de las atracciones disponibles y luego hacer clic sobre una atracción para poder ver su detalle y su ubicación en un mapa.

Le pedimos al diseñador que nos enviara un diseño en principio para un dispositivo móvil y veremos que después también podremos generar la aplicación en Angular usando los mismos objetos creados en la importación.

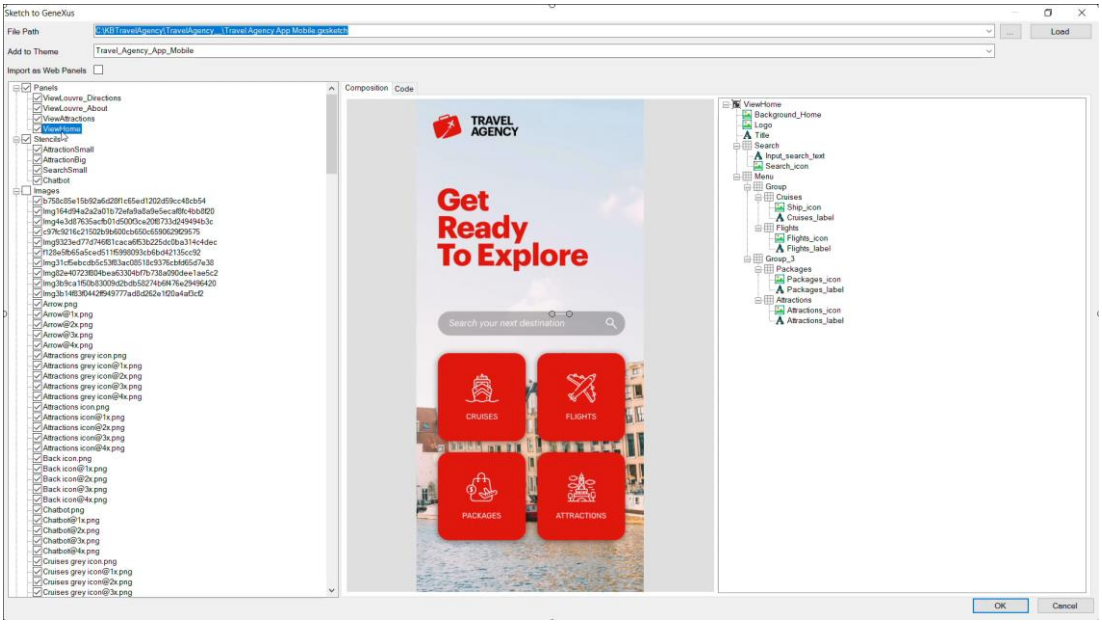
Importación desde Sketch



Para cumplir con lo que le pedimos, el diseñador creará un diseño mediante la herramienta Sketch y luego utilizará un plugin de GeneXus instalado en la misma que permite crear un archivo de extensión gxsketch, que es el que nos envía.

Lo que hacemos a continuación es importarlo en nuestra KB y se integrarán todos los componentes de diseño e incluso se crearán los objetos GeneXus correspondientes para albergar las definiciones.

Importación del archivo .sketch



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

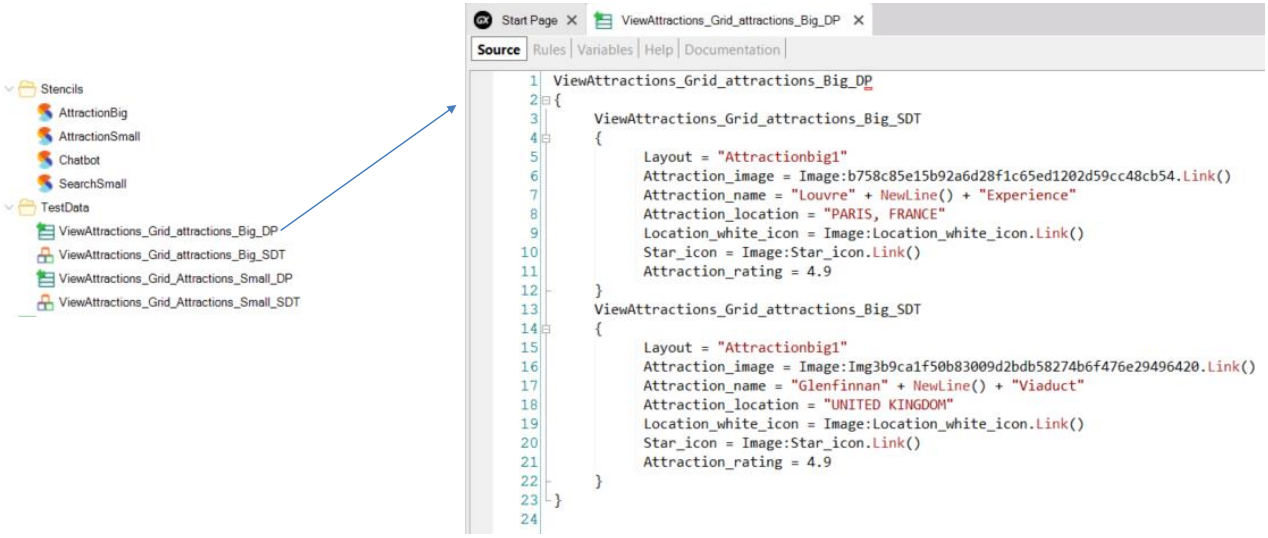
El check box Import as Web Panels lo dejamos desmarcado, porque queremos que nos cree objetos panel y no webpanels.

Si hacemos clic en 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 y mostrar atracciones cargadas, que luego deberemos sustituir por las almacenada en la base de datos de nuestra aplicación.

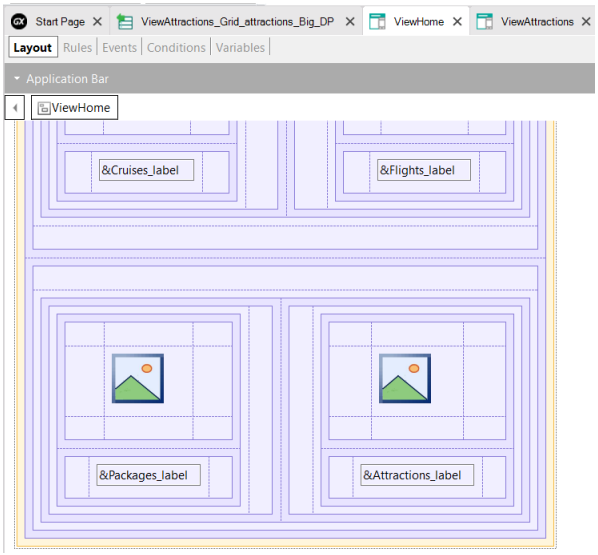
Y más abajo vemos los fuentes que se utilizaron en el diseño. Estamos de acuerdo, así que presionamos OK.

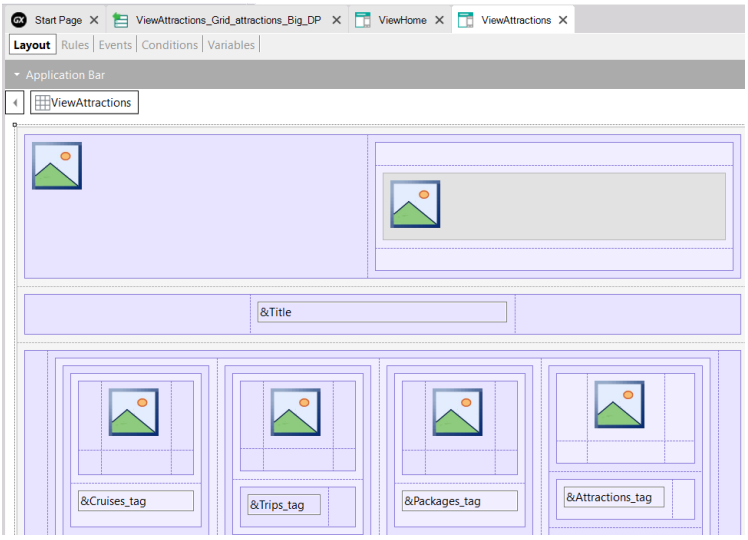
Objetos creados en la importación



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 con algunos stencils utilizados para encapsular diseño en la aplicación y Test Data, que si lo abrimos vemos que contiene data providers y sdt para cargar datos fijos.
Si ahora abrimos el primer data provider vemos el código de carga de las imágenes.

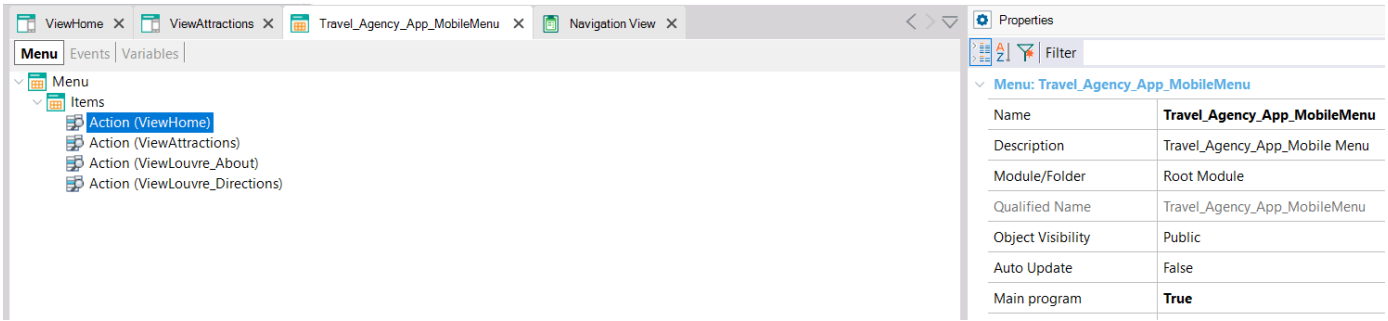
Objetos panels creados automáticamente en la importación





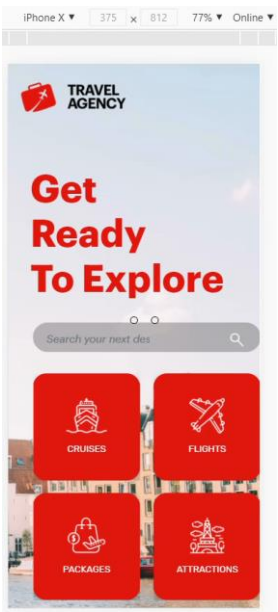
Vemos que se crearon automáticamente los objetos panels que habíamos visto en la pantalla del wizard de importación, como el de la página inicial (ViewHome) y también el ViewAttractions y los invocados por él, ViewLouvre_About y ViewLouvre_Directions.
Si vemos el form del ViewHome observamos que ya están todos los componentes creados para el contenido visual, lo mismo para el ViewAttractions.

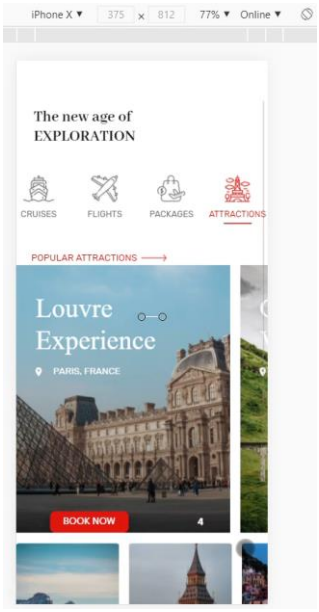
Objeto main de nuestra aplicación



Aquí podemos ver que se también creó un objeto menú que será el objeto main que vamos a ejecutar.
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 primera opción del menú.
Para ejecutar, ponemos a esto objeto como Startup Object y damos F5.

La aplicación en ejecución con datos fijos



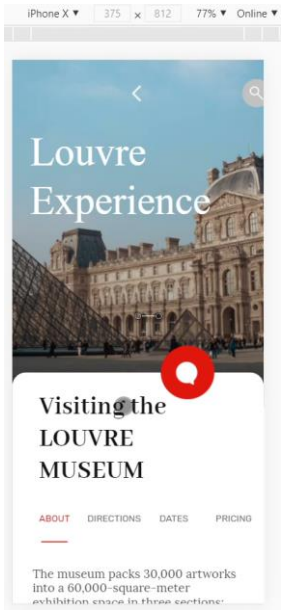


Como nuestro diseño fue hecho en principio para plataforma mobile, elegimos un tamaño de pantalla para iPhone X. Luego tendremos que hacer los cambios necesarios para poder verlos en una pantalla para desktop.

Vemos que se ejecuta la pantalla inicial con botones a la distintas partes de nuestra aplicación. Hacemos clic en Attractions y vemos la lista de atracciones populares y más abajo otras atracciones para visitar.

Recordemos que estamos viendo los datos fijos que cargaron los data providers de la importación.

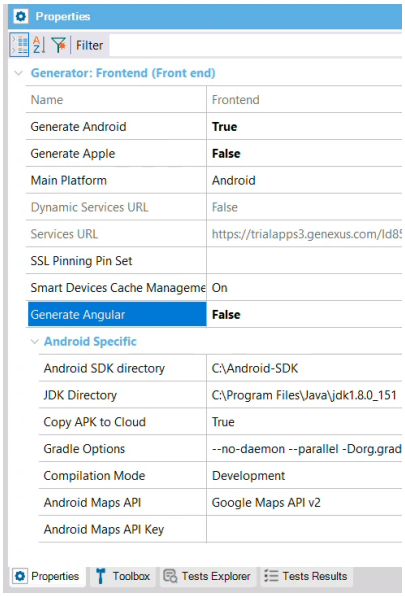
La aplicación en ejecución con datos fijos (cont.)



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

Si ahora hacemos clic en Directions, se abre el panel con los datos de la dirección del Louvre y un mapa con su localización.

Generación y ejecución de la aplicación customer-facing en Android



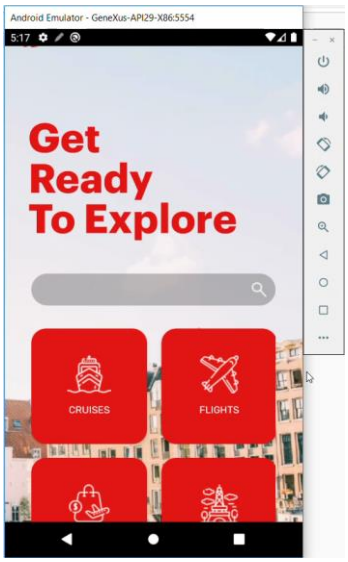
Ahora generaremos la aplicación en Android para ver cómo queda con el nuevo diseño en una plataforma mobile.

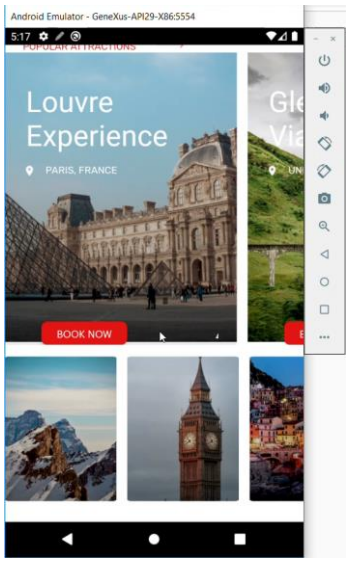
Para eso vamos a la propiedades del Front end y elegimos Generate Angular en True.

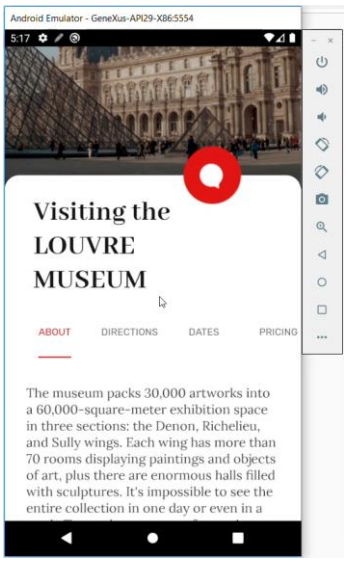
Como queremos ver ahora solamente la aplicación mobile, vamos a poner la propiedad Generate en False.

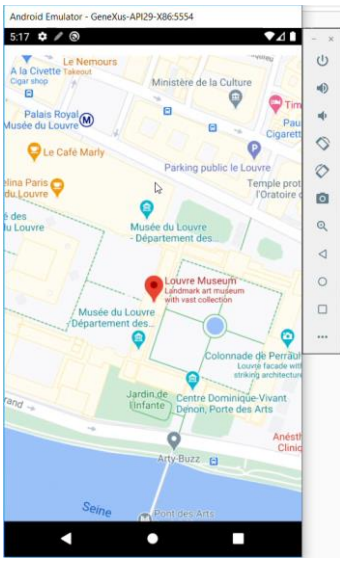
Como ya habíamos puesto al objeto menú como Startup Object, presionamos F5.

Generación y ejecución de la aplicación customer-facing en Android





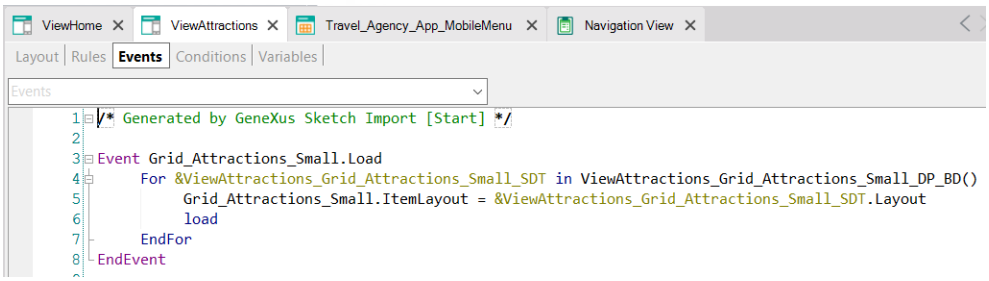




Vemos que se abre el emulador y la aplicación se ejecuta en Android, con la visualización estética que esperábamos, acorde al Design System definido por nuestros diseñadores.

Este ejemplo nos permitió ver cómo es el proceso de desarrollo de una aplicación customer-facing y cómo es importante 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



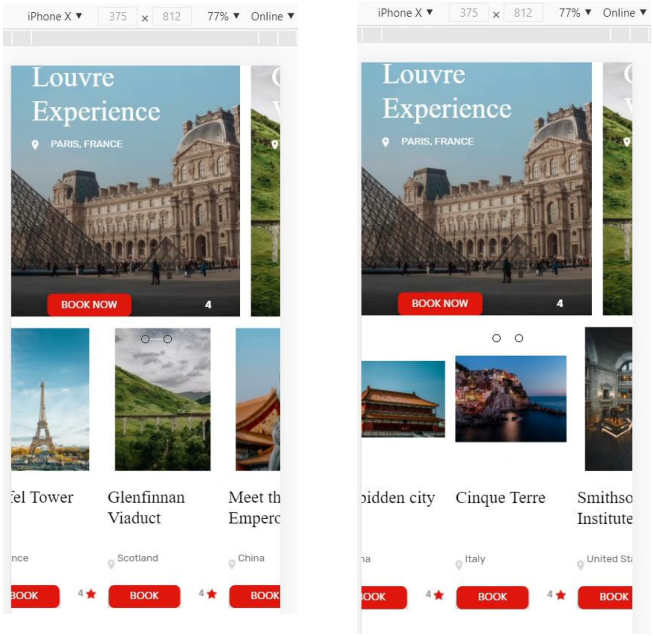
Hasta ahora estuvimos viendo datos fijos de prueba, que los diseñadores agregaron para que podamos ver el comportamiento de la aplicación con el nuevo diseño. Así que vamos a hacer algunos cambios, para que podamos ver los datos reales de nuestra base de datos.

Si vamos al panel ViewAttractions, vemos que la grilla muestra las atracciones que están cargadas en una variable tipo SDT de nombre ViewAttractions_Grid_Attractions_Small_SDT. Si vamos a los eventos del panel, vemos que ésta es cargada con un data provider. Lo abrimos, hacemos Save As y guardamos el nuevo data provider con el nombre ViewAttractions_Grid_Attractions_Small_ 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 y a Attraction_location le asignamos el atributo CountryName. 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 y vamo nuevamente a habilitar la generación en Angular. Ahora damos botón derecho sobre nuestro objeto main y elegimos Run.

Ejecución de la aplicación con datos reales

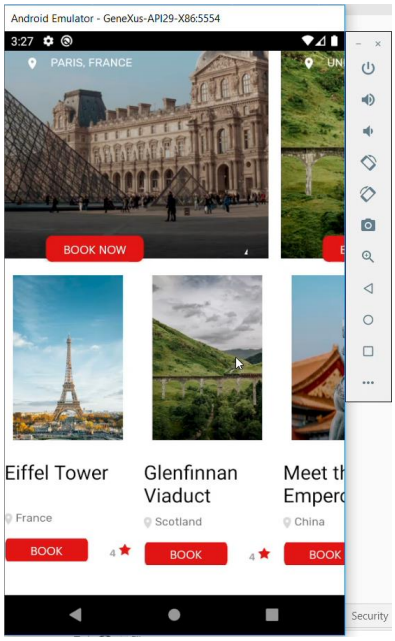


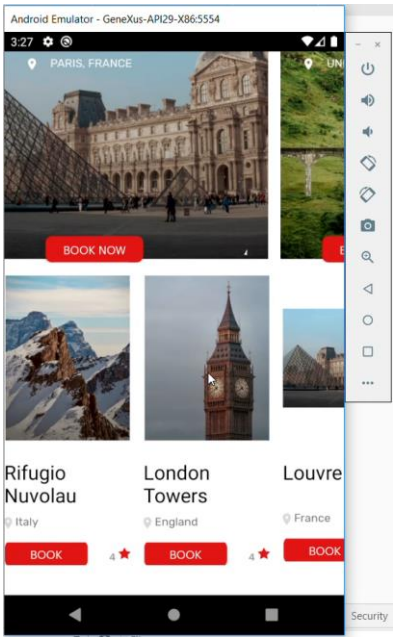
Si vamos a las atracciones, vemos que estamos recorriendo las atracciones que teníamos cargadas....pero ahora con la aplicación con un diseño más apropiado.

No cabe duda que si disponemos de un diseñador en nuestro equipo, nunca haríamos el trabajo de modificar las clases y propiedades a mano, 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.

Para terminar, vamos a generar en Android para ver cómo se ve el resultado final con los datos de la base de datos en el dispositivo móvil.

Ejecución de la aplicación con datos reales





Y también vemos como la aplicación en Android está mostrando todas las atracciones que tenemos en la base de datos y también con el nuevo diseño.

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.

Esta forma de trabajar, donde el diseño es parte del desarrollo, queda plasmada en la aplicación de la metodología DevOps, que veremos más adelante.

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