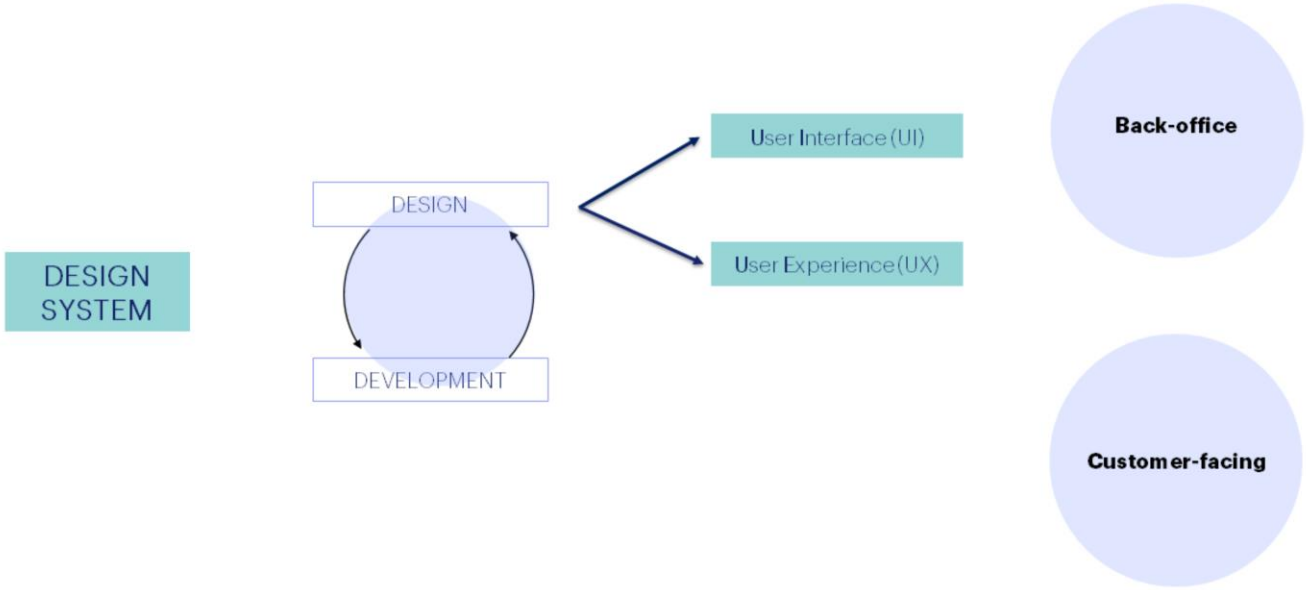
Web screens focused on Customer-facing

Design System

GeneXus™

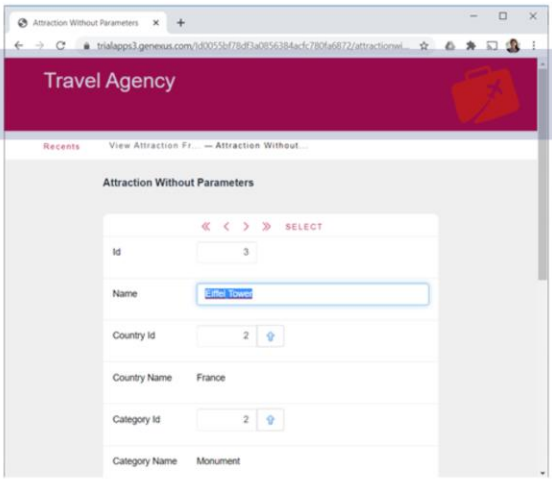
So far, we have not considered the design aspect of our panels, for we have been focusing on their functioning.
Let's now see how to include design in our application.

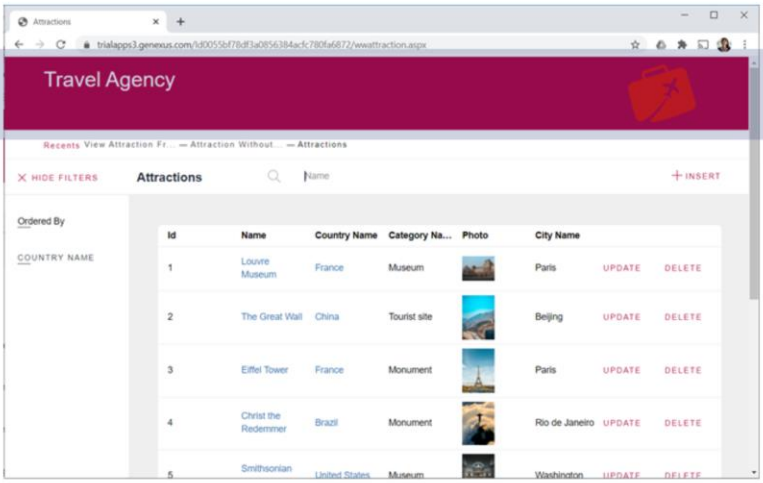


In previous videos, we saw that both back-office apps and customer-facing apps imply design and usability requirements that introduced the concept of Design System.

The Design System involves a group of Principles, Patterns and Practices that convey consistency, uniformity and robustness to your app, defining a development cycle that includes both developers and designers.

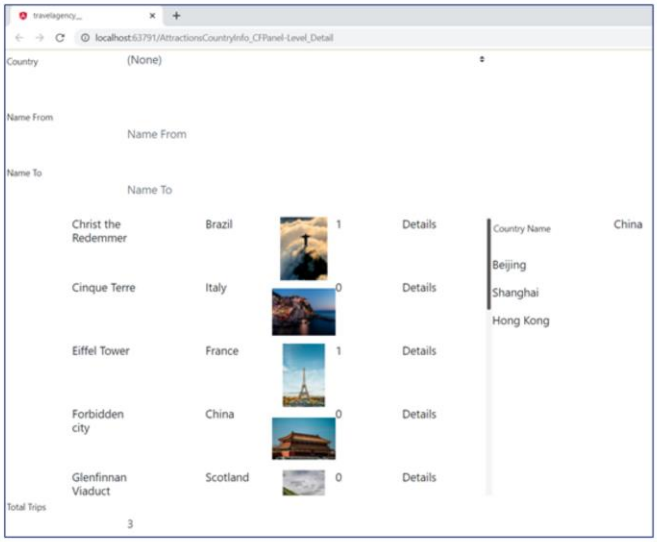
Predefined Design System in a web app

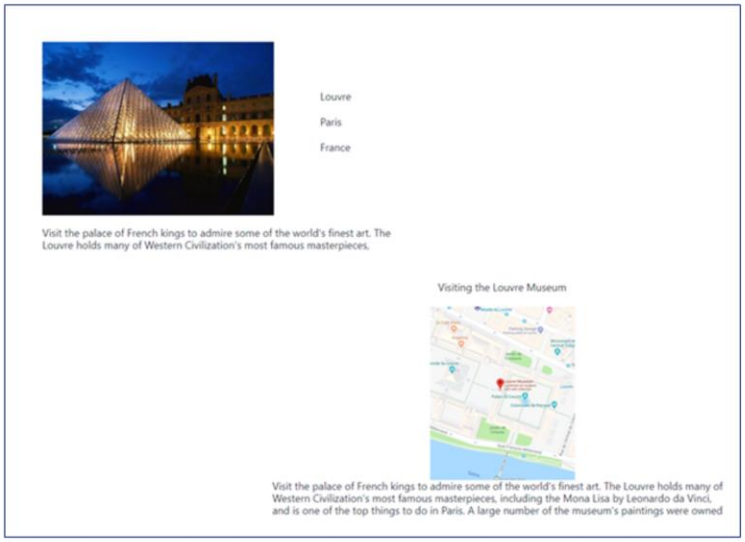




We also saw that the transactions, as well as the webpanels, that were part of the back-office, had a predefined Design System.

Predefined Design System in customer-facing



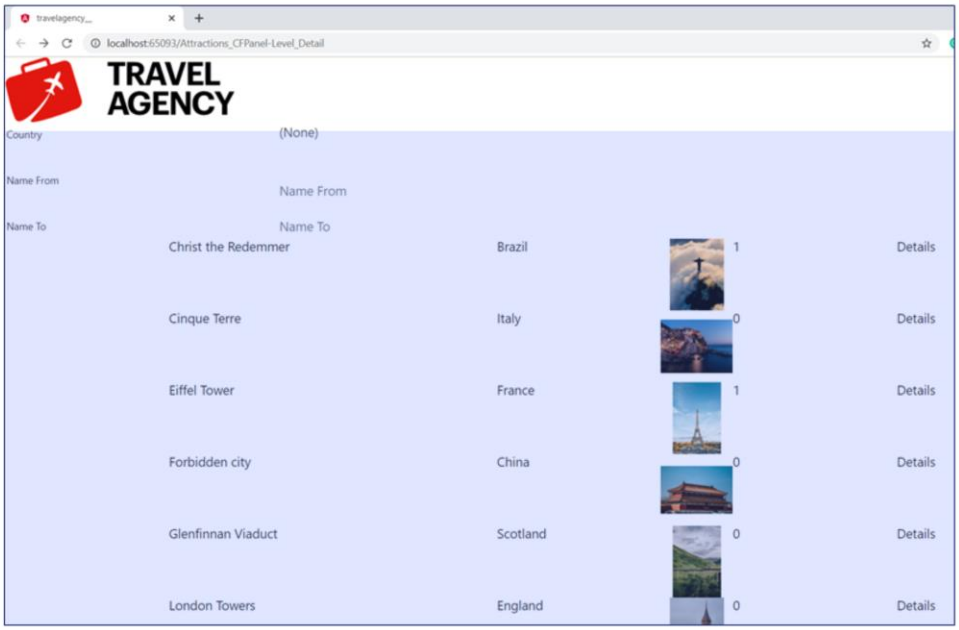


Just like in the case of the back-office app, the objects you use in the customer-facing part also have a predefined design that GeneXus includes there.

As opposed to web panels and transactions objects –which had a master page assigned by default– the objects in the front-end customer-facing do not have an assigned Master Panel object assigned by default to provide a container to execute the app and include basic design components in the execution.

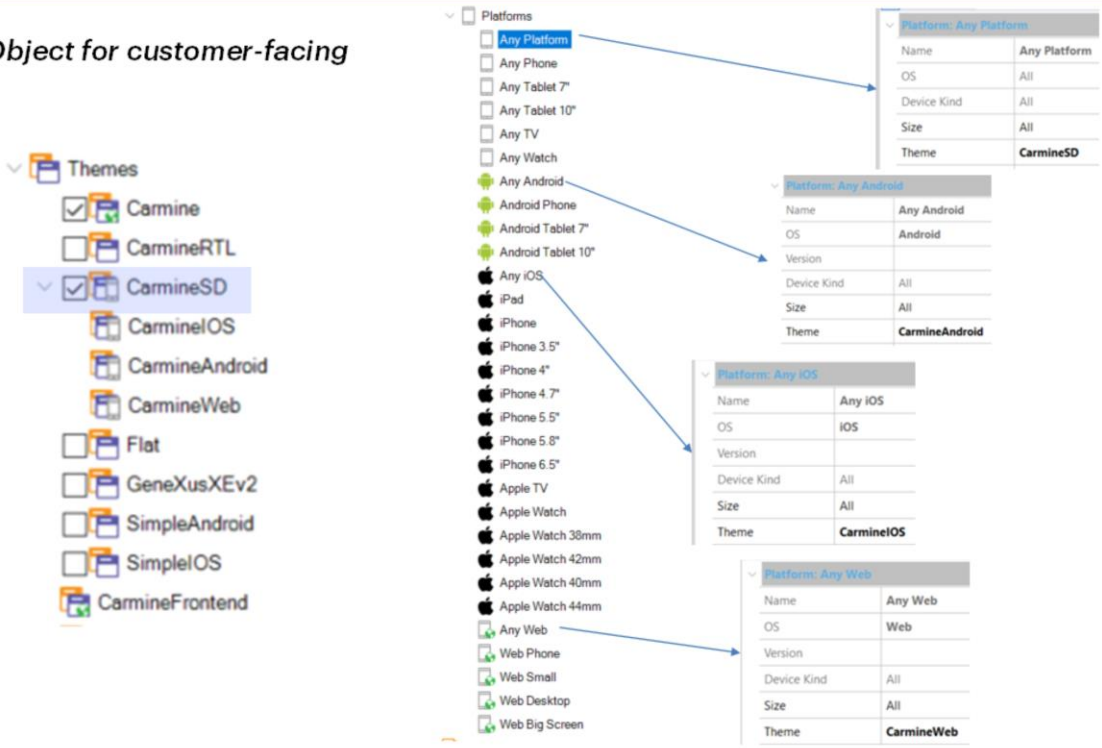
Anyhow, you will see that the controls abide by a design and they are consistent in what concerns their appearance -all this in a predefined manner. We also saw that it is possible to assign a Master Panel object to the app's panels.

Predefined Design System in customer-facing



We can see that the controls abide by a design and they are consistent in what concerns their appearance -all this in a predefined manner. We also saw that it is possible to assign a Master Panel object to the app's panels.

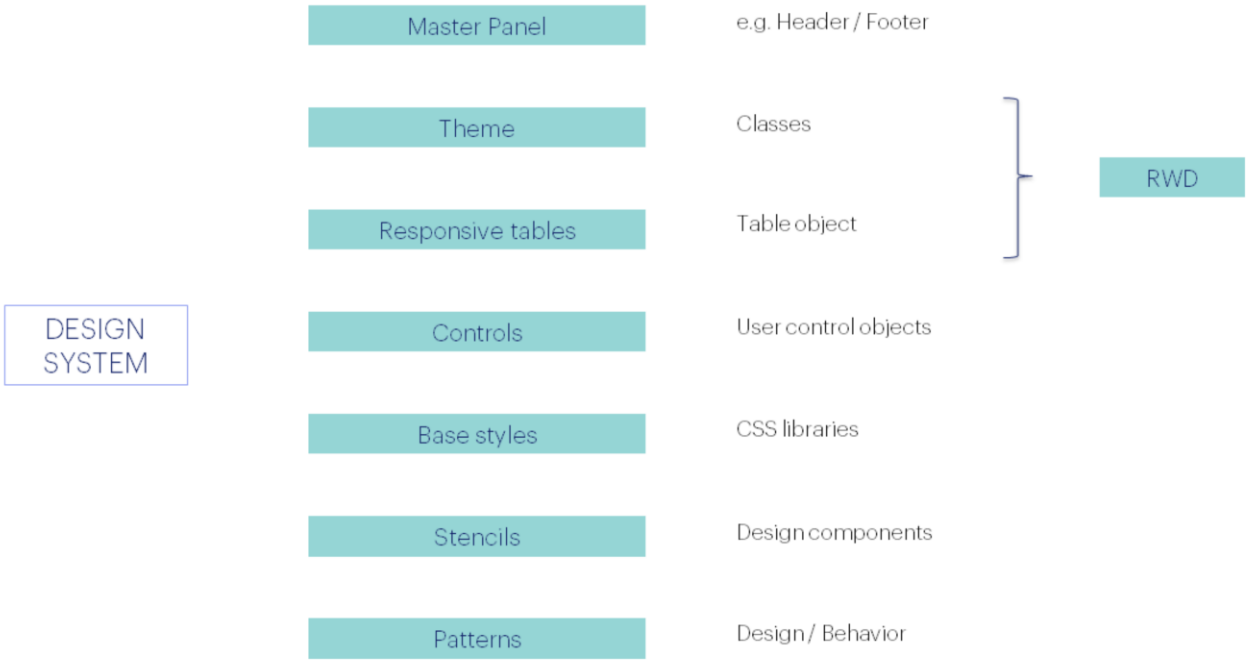
Theme Object for customer-facing



The appearance of controls on screen is considered on the basis defined for the Theme object called CarmineSD.

Then one of the subthemes is used, depending on the platform selected. For example, when generating in Android we use CarmineAndroid, for Apple we use CarmineIOS, and in Angular generation we take the definitions of the CarmineWeb theme.

You may verify this by opening the Platforms node in the KB Explorer; for each platform you will see its Theme property. For example, for Any Platform you will see that the CarmineSD theme has been assigned, while for Any Android the theme assigned is CarmineAndroid; for Any iOS you will see the CarmineIOS theme, and for Any Web the CarmineWeb theme.



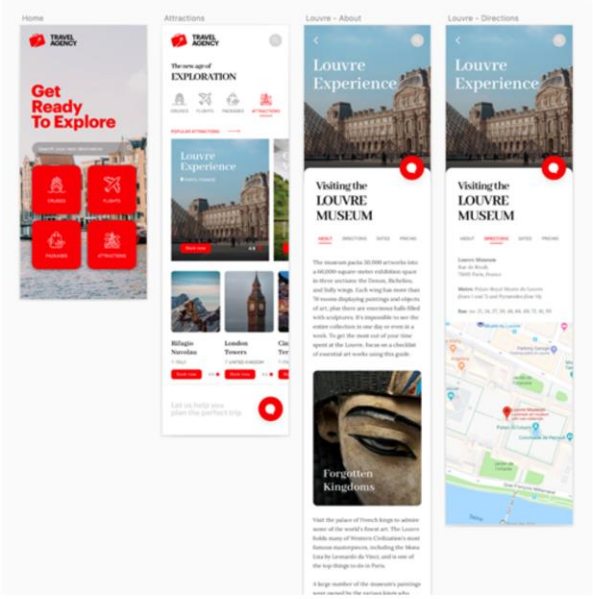
Regarding the components that we mentioned as being part of a Design System: in a customer-facing app, you also have a master panel, theme objects and classes. There isn't one control specific for responsive tables, because the Table controls enables the automatic adjustment of contents when a web app is generated with Angular.

Unlike the Work With pattern for web that generates web panel objects, the Work With pattern for customer-facing objects does not generate panel objects but one specific object instead, called WorkWithDevices.

All the other Design System components are represented, besides the fact that the implementation of each varies according to the platform (web or native) and with the generator used (Angular, Android, or Apple).

Similarly to what we saw in the design customization for the back-office app, in customer-facing apps, you may work with the various elements in the Design System (classes, themes, controls, etc.) to achieve a design that will enable the best user experience possible.

Importing a design from Sketch

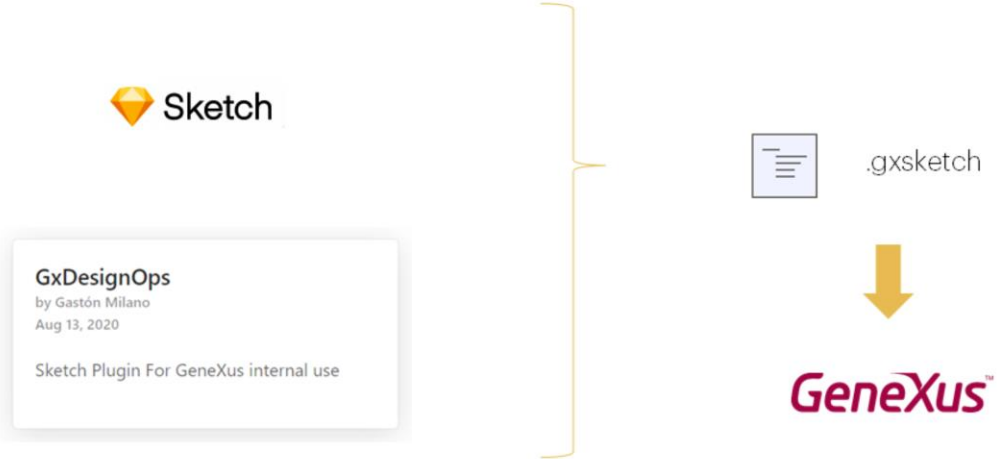


However, the simplest and best way to go about it is for a designer to define all that for you to then import the design into your KB.

You will want to create an initial page with buttons to the different parts of the app, but if you're interested only in viewing the data relative to tourist attractions, you will need a button to show a list of attractions available to then click on one attraction to see details and a location map.

If you ask the designer to send you a design for a mobile device in the first place, you will later be able to generate the app in Angular, using the same objects created in the import.

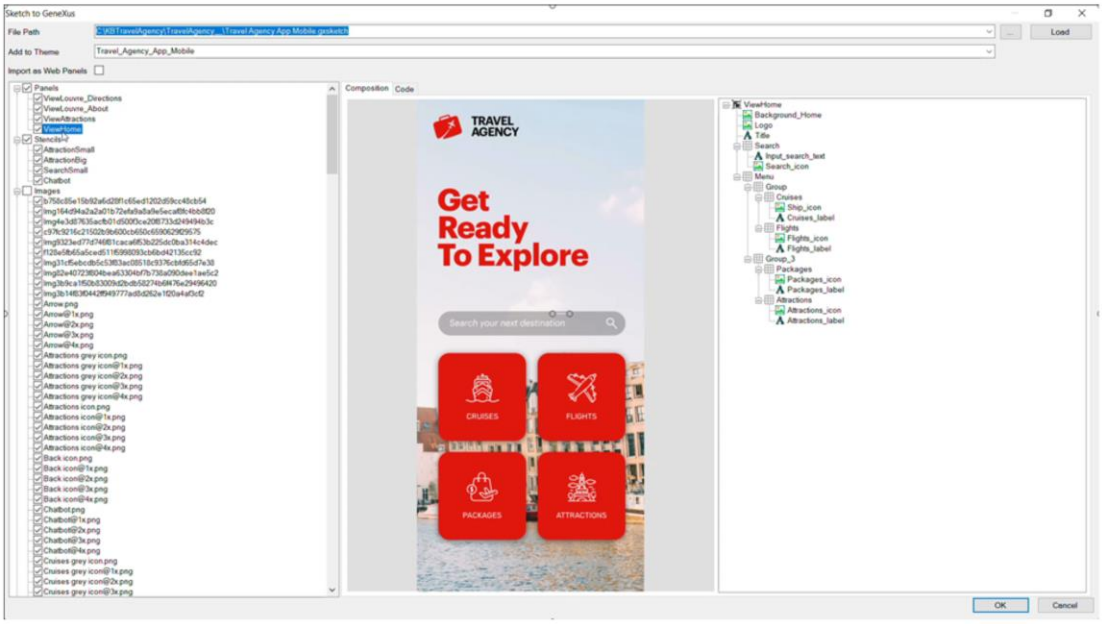
Importing from Sketch



In response to your requirement, the designer will create a design with the Sketch tool and then he/she will use a GeneXus plugin installed in this tool, that allows the creation of a file with extension gxsketch that you will receive.

Next, what you must do is import it to your KB to integrate all the design components and also create the corresponding GeneXus objects to contain the definitions.

Importing the .sketch file



To import the .sketch file sent by the designer you should go to Tools/Application Integration, and select Sketch import.

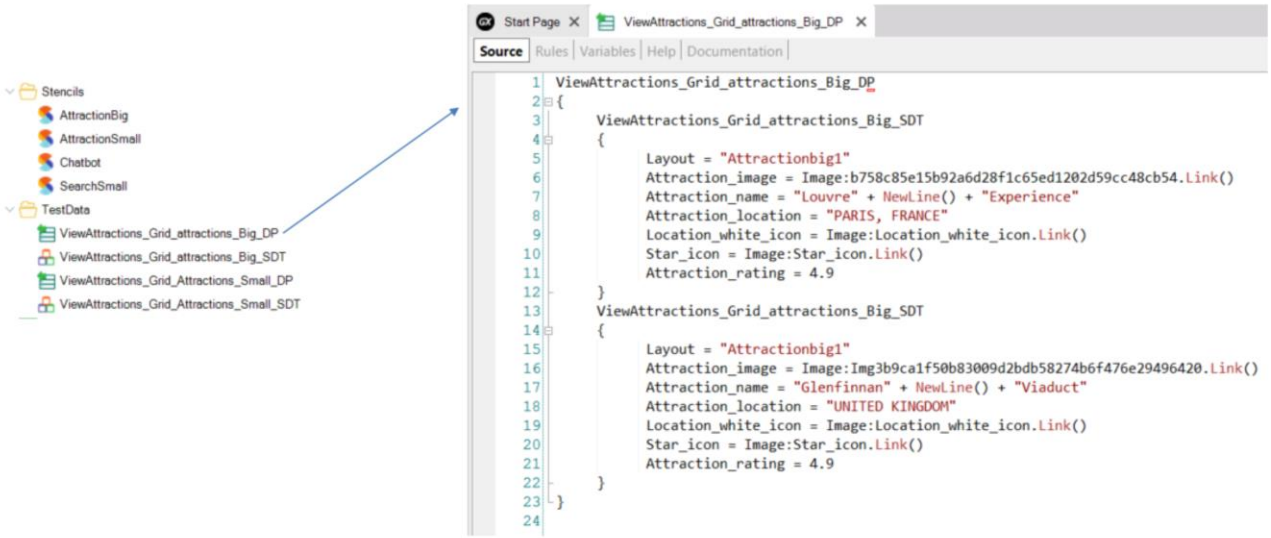
Uncheck the checkbox called Import as Web Panels, because you need to create panel objects and not web panels.

If you click on the panels to be created by the import, you will see a preview where you may verify that they correspond to the design validated. To the right, you have information about the controls it will contain.

You will see that it also imports images to execute the panel with fixed data and show the attractions loaded, which you will later have to substitute by those saved in your app's database.

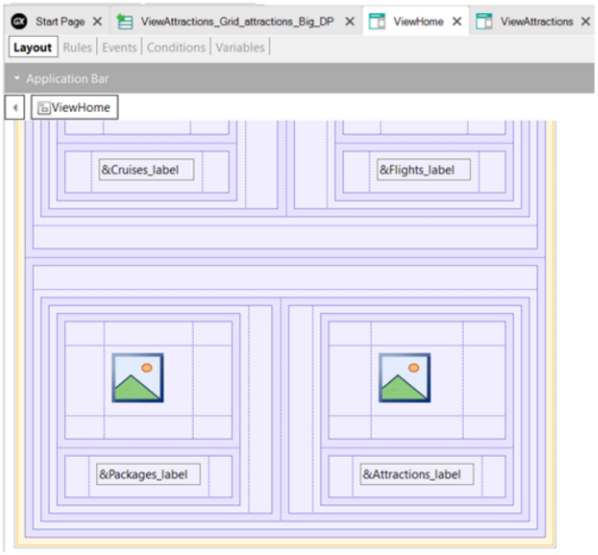
Further down you will see the fonts used in the design. If you agree, press OK.

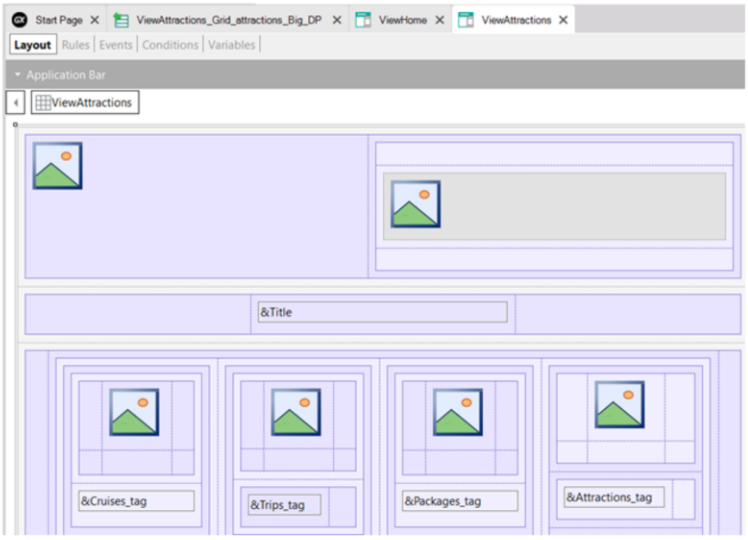
Objects created in the import



The Output window shows the import's progress and the outcome, free of errors.
In KB Explorer you will see that two folders were created, namely: Stencils, with some stencils used to encapsulate design in the app, and Test Data, which you may open to see that it contains data providers and sdt to load fixed data.
If you now open the first data provider you will see the image load code.

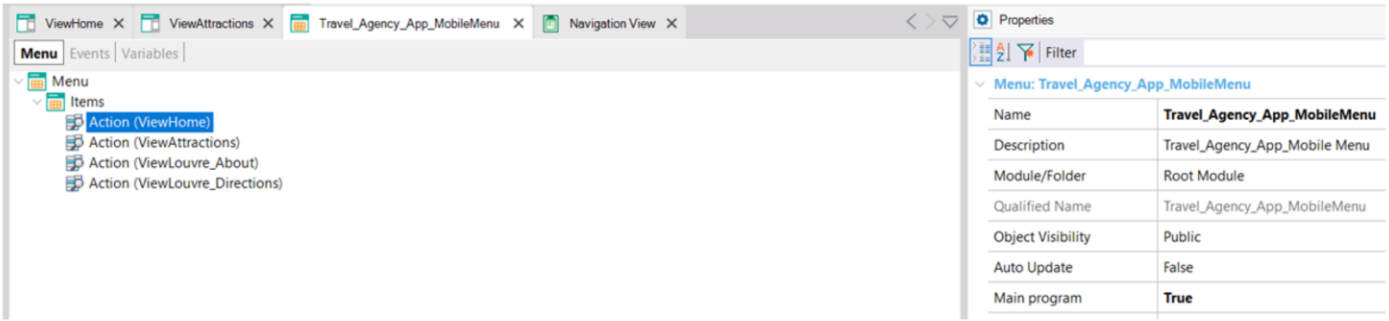
Panel objects automatically created in the import





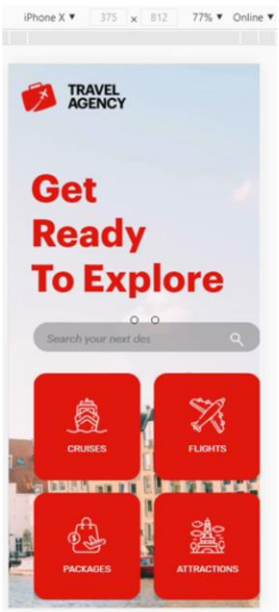
You will see that, the panel objects you saw in the import wizard screen were created automatically, as well as the one in the initial page (ViewHome) and the ViewAttractions and those invoked by it (ViewLouvre_About, and ViewLouvre_Directions).
If you take a look at the form of ViewHome, you will see that all components have been created for the visual content, and the same happens with ViewAttractions.

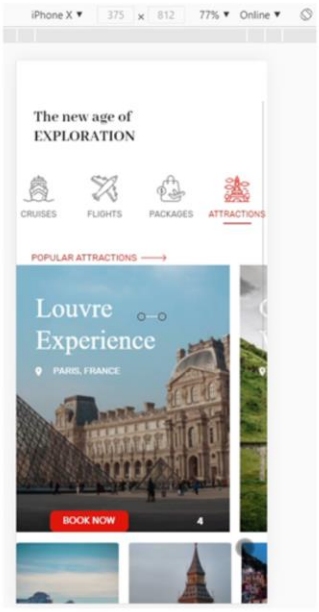
Main object in the app



Here, you will see that a menu object was also created . It will be the main object that you will execute.
This object is seen only in the generation for native platform. In the case of Angular, the first object you see is ViewHome, the first option in the menu.
For the execution, set this object as Startup Object and press F5.

The app in runtime, with fixed data



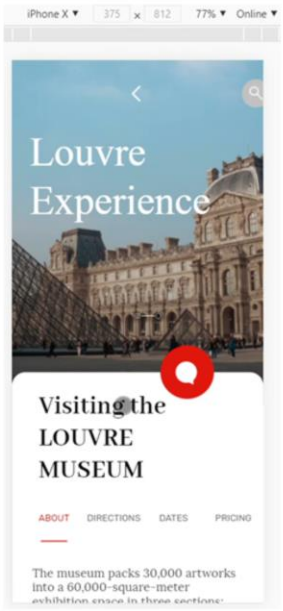


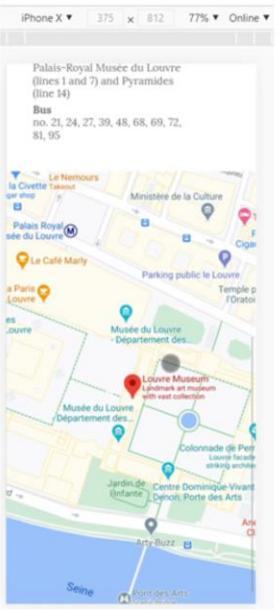
Since the design was initially meant for the mobile platform, select an iPhone X screen size. Later, you will have to implement the changes necessary to view on a desktop screen.

The initial screen is executed with buttons to the various parts of the app. By clicking on Attractions, you will see the popular attractions list and, below that, other attractions that may be visited.

You should recall that you have at sight the fixed data loaded by the import's data providers.

The app in runtime, with fixed data (continued)

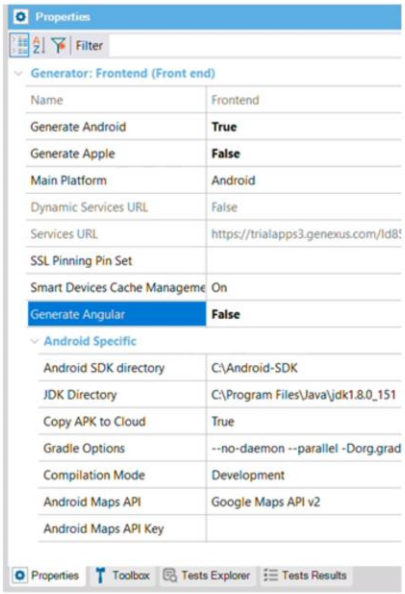




When you click on Louvre, you will get a page with detailed information, within a neatly organized design that is aesthetically agreeable.

If you click on Directions, the panel opens up with the details of the Louvre museum's address and a location map.

Creation and execution of the customer-facing app in Android



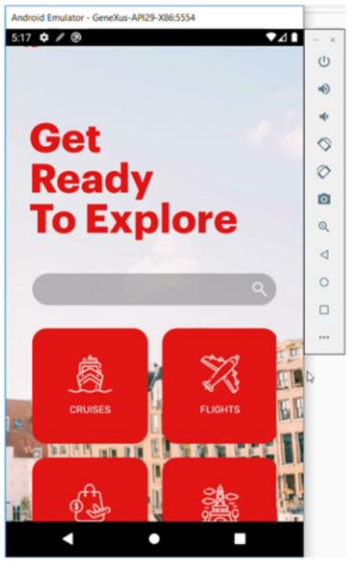
Now you will generate the app in Android to see what it looks like with the new design, on a mobile platform.

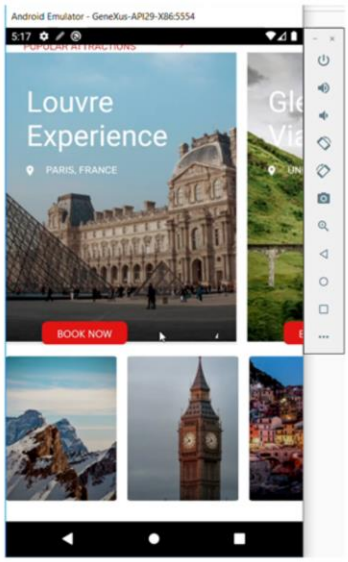
To do that, go to the Front end properties and select Generate Angular as True.

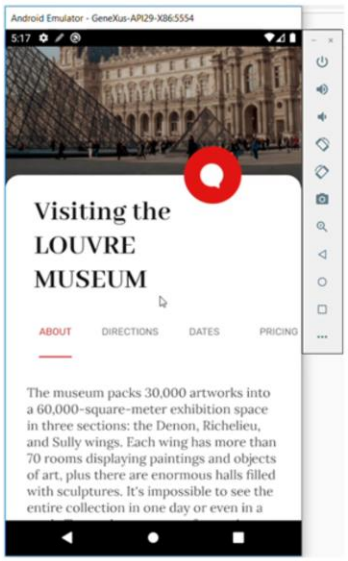
Since you only want to see the mobile app now, set the Generate property as False.

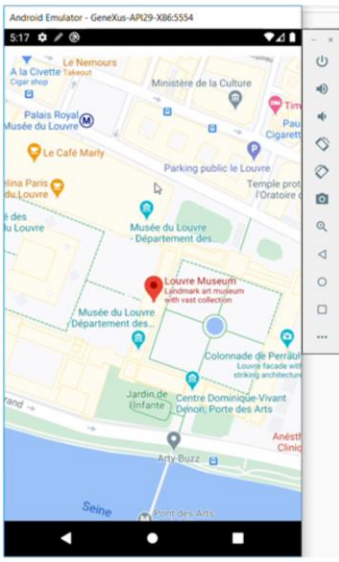
The menu object was already established as Startup Object, so, just go on to press F5.

Creation and execution of the customer-facing app in Android





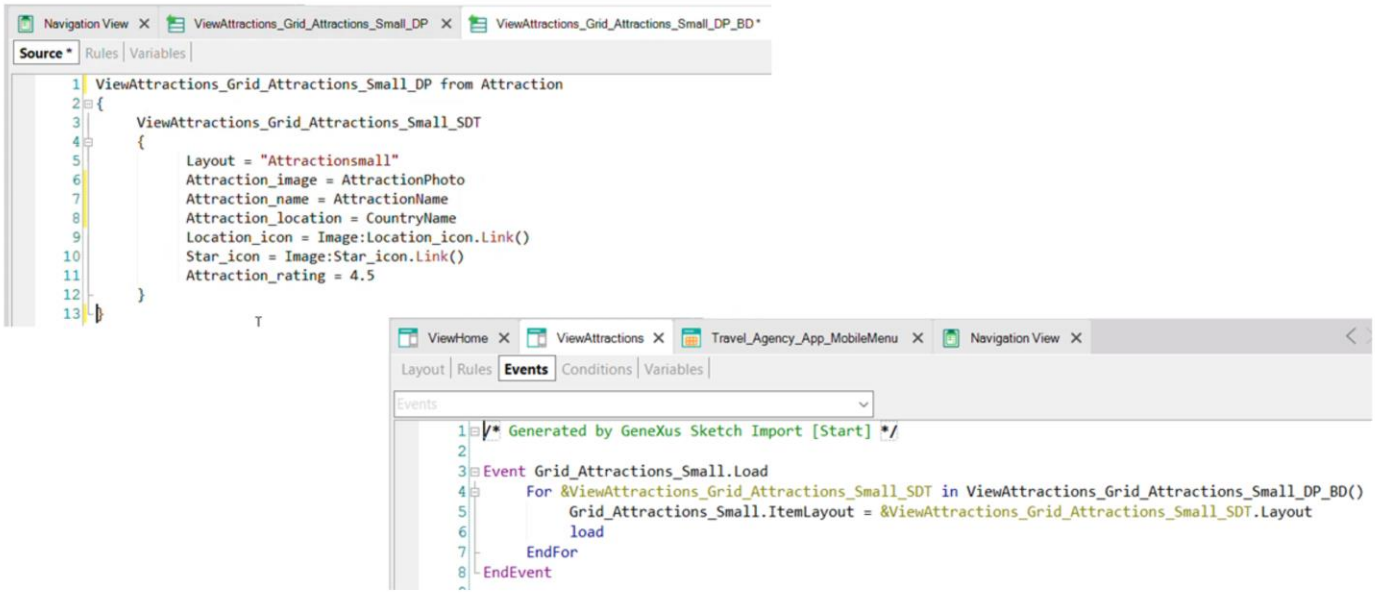




You will see that the emulator opens up and the app is executed on Android, with the aesthetic view expected, in accordance with the Design System defined by the designers.

This example has shown the development process for a customer-facing application, and the significance of team work with individuals of varied profiles, where each will contribute to the best solution possible.

Modification of new objects for accessing data from the database



To this point, you have seen fixed data for testing that the designers have added to show the app's behavior with the new design. Now, you will make some changes to see the actual data from the database.

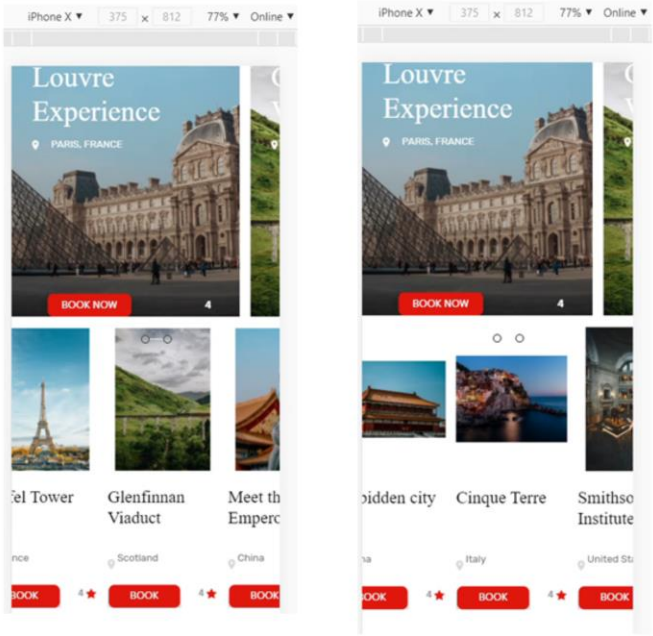
If you go to the ViewAttractions panel, you will find that the grid shows the attractions loaded in a variable of the SDT type, called ViewAttractions_Grid_Attractions_Small_SDT. In the panel's events, you will see that it is loaded with a data provider. Open it, do Save As and save the new data provider under the name: ViewAttractions_Grid_Attractions_Small_ DP_BD.

Add the from Attractions clause and load the Attraction_image field with the value of the AttractionPhoto attribute, the Attraction_name field with the AttractionName attribute, and to Attraction_location assign the CountryName attribute. Then delete everything that is not necessary.

Go back to the ViewAttractions panel and change the data provider that loads the attractions for the new one you just built.

Save and enable the generation in Angular again. Now right click on the main object and select Run.

Running the app with actual data

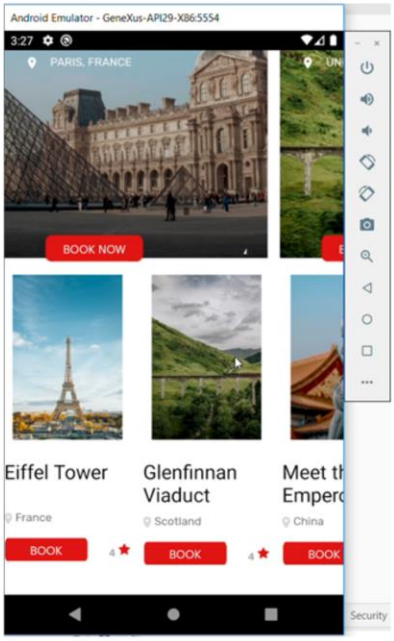


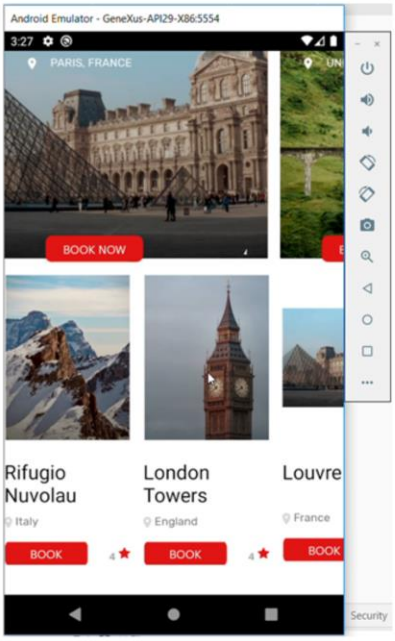
When you go to attractions, you will see that you're going over the attractions that were loaded...but now the app has a more adequate design.

Undoubtedly, when we have a designer at hand in our team, we would never go into the work of modifying classes and properties manually, because importing these definitions from Sketch is a more professional solution for the design, and we can focus on the more functional aspects of the app.

To conclude, let's now generate in Android so you can see the final outcome with the database data on the mobile device.

Running the app with actual data





In Android, you can also see that the app shows all the attractions from the database, with the new design.

This method, where the design of a professional is integrated into your GeneXus project implies several advantages, because efforts are optimized and each member of the team contributes according to his/her own profile, with what he/she does best.

The work method where design becomes part of the development activity is applied in the DevOps methodology which will be shown in videos to come.

GeneXus™

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