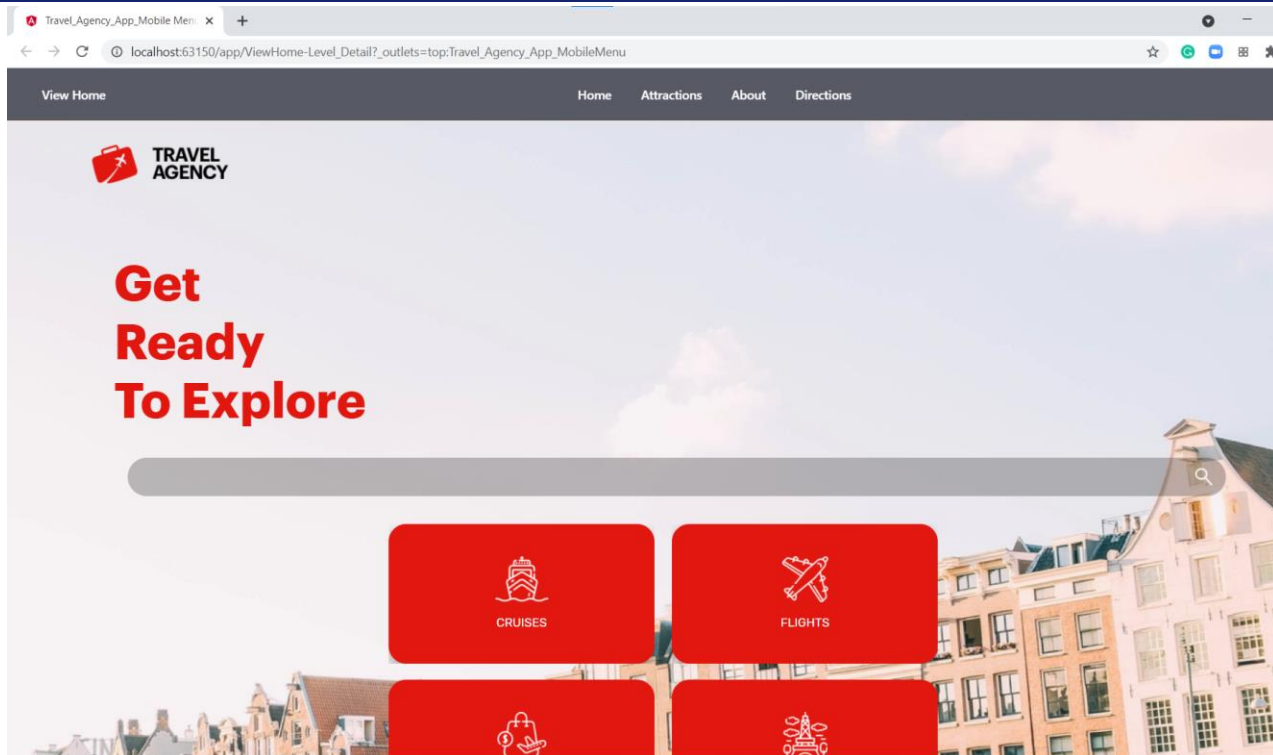


Primeiros passos com Angular

GeneXus™

No desenvolvimento do curso mostraremos como implementamos uma aplicação para os clientes de uma Agência de Viagens e faremos todos os nossos exemplos completando o seu desenvolvimento. Mas primeiro vejamos a aplicação já em funcionamento.



Abrimos GeneXus, pressionamos F5 e vemos que se abre o navegador web onde vemos a tela inicial da aplicação.

Na parte superior vemos o logotipo da agência e na parte cinza uma barra com botões de acesso rápido aos dados das atrações. No centro da página, vemos grandes botões que nos permitem acessar as diferentes partes da aplicação para visualizar informações sobre cruzeiros, voos, pacotes turísticos e atrações.

The screenshot shows a mobile application interface for 'View Attractions'. At the top, there is a navigation bar with a back arrow, the text 'View Attractions', and menu items: 'Home', 'Attractions', 'About', and 'Directions'. Below the navigation bar, the main heading reads 'The new age of EXPLORATION'. A horizontal menu contains four icons: 'CRUISES' (ship), 'FLIGHTS' (airplane), 'PACKAGES' (hand holding a document), and 'ATTRACTIONS' (crown), with 'ATTRACTIONS' being the active selection. Below this menu, a section titled 'POPULAR ATTRACTIONS' with a right-pointing arrow lists five items in a horizontal scroll view:

- Louvre Experience**: PARIS, FRANCE. Rating: 4 stars. Button: BOOK NOW.
- Glenfinnan Viaduct**: UNITED KINGDOM. Rating: 4 stars. Button: BOOK NOW.
- Eiffel Tower**: PARIS, FRANCE. Rating: 5 stars. Button: BOOK NOW.
- Glenfinnan Viaduct**: UNITED KINGDOM. Rating: 4 stars. Button: BOOK NOW.
- Louvre Experience**: PARIS, FRANCE. Rating: 4 stars. Button: BOOK NOW.


Se clicamos neste último botão, acessamos uma página onde podemos navegar horizontalmente entre as atrações mais populares. Para cada atração, vemos sua imagem, sua localização, sua qualificação e um botão para agendar uma visita.

Travel_Agency_App_Mobile Men x +

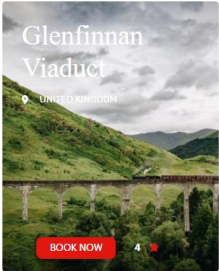
localhost:63150/app/ViewAttractions-Level_Detail?_outlets=top:Travel_Agency_App_MobileMenu

View Attractions Home Attractions About Directions


POPULAR ATTRACTIONS →



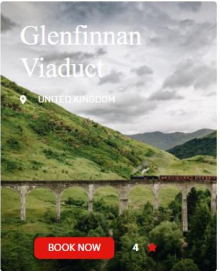
Louvre Experience
PARIS, FRANCE
BOOK NOW 4 ★




Glenfinnan Viaduct
UNITED KINGDOM
BOOK NOW 4 ★




Eiffel Tower
PARIS, FRANCE
BOOK NOW 5 ★




Glenfinnan Viaduct
UNITED KINGDOM
BOOK NOW 4 ★




Louvre Experience
PARIS, FRANCE
BOOK NOW 4 ★




Louvre Experience
PARIS, FRANCE
BOOK NOW 4 ★




Rifugio Nuvolau
ITALY
BOOK NOW 5 ★




London Towers
UNITED KINGDOM
BOOK NOW 4 ★




Cinque Terre
ITALY
BOOK NOW 4 ★




Sacre Coeur
FRANCE
BOOK NOW 5 ★



Statue of Liberty
USA
BOOK NOW 4 ★

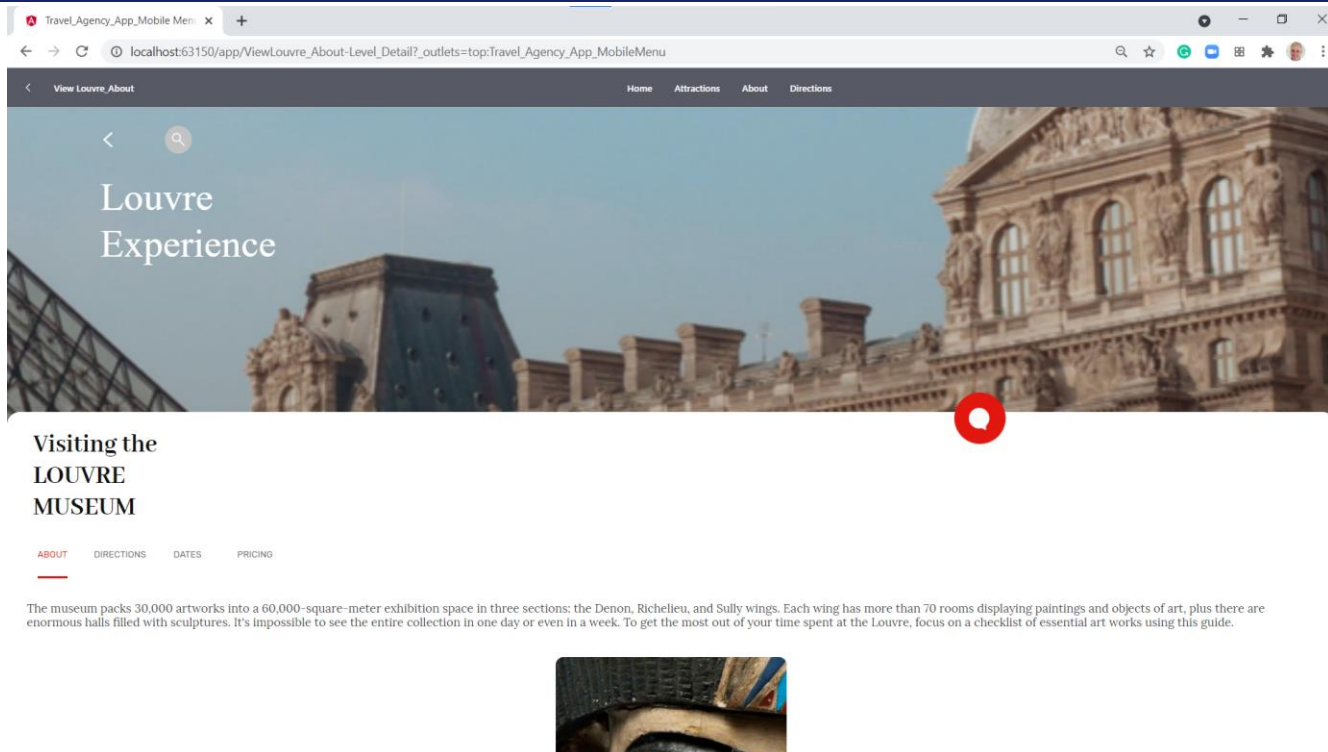


Taj Mahal
INDIA
BOOK NOW 4 ★



Chichen Itza
MEXICO
BOOK NOW

Mais abaixo na página, vemos outra lista com todas as atrações para visitar. Aqui também podemos nos mover horizontalmente e para cada atração vemos seu país, qualificação e botões para marcar uma visita.



Se clicamos em uma atração popular, por exemplo, no Louvre, vemos que se abre uma página com o detalhe da atração, onde podemos ler uma breve descrição com imagens representativas.

Vemos também um submenu, onde além destes dados gerais (marcado com About) podemos acessar outros dados da atração como seus endereços, datas das visitas e seus preços.

The screenshot shows a mobile application interface for the Louvre Museum. At the top, there is a browser address bar with the URL "localhost:63150/app/ViewLouvre_Directions-Level_Detail?_outlets=top:Travel_Agency_App_MobileMenu". Below the browser bar is a navigation bar with the title "View Louvre_Directions" and menu items: "Home", "Attractions", "About", and "Directions". A red speech bubble icon is visible in the top right corner of the page.

Visiting the LOUVRE MUSEUM

ABOUT DIRECTIONS DATES PRICING

Louvre Museum
Rue de Rivoli,
75001 Paris, France

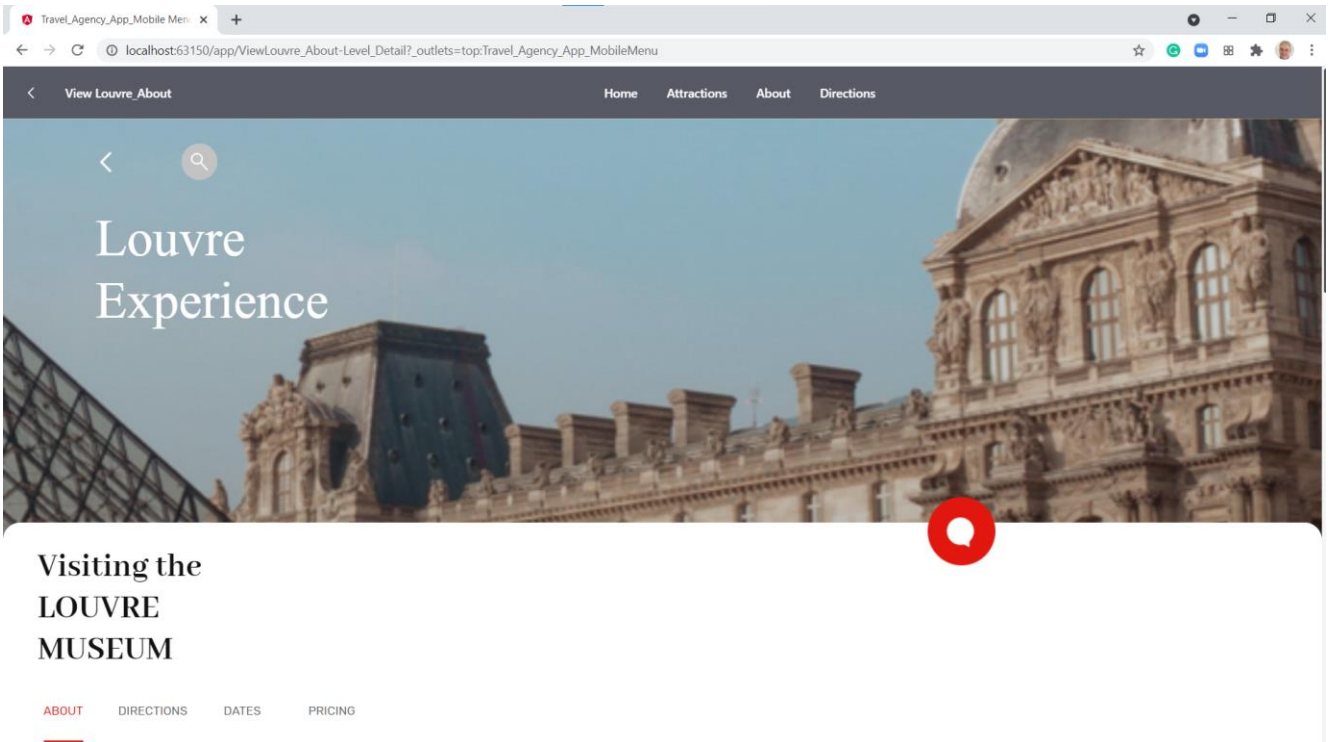
Metro
Palais-Royal Musée du Louvre
(lines 1 and 7) and Pyramides (line 14)

Bus
no. 21, 24, 27, 39, 48, 68, 69, 72, 81, 95

The bottom half of the screenshot shows a map of the Louvre Museum area. A red location pin is placed on the map, labeled "Louvre Museum Landmark art museum". Other labels on the map include "Société des Amis du Louvre", "Musée du Louvre - Département des...", and "Louvre - Rivoli M".

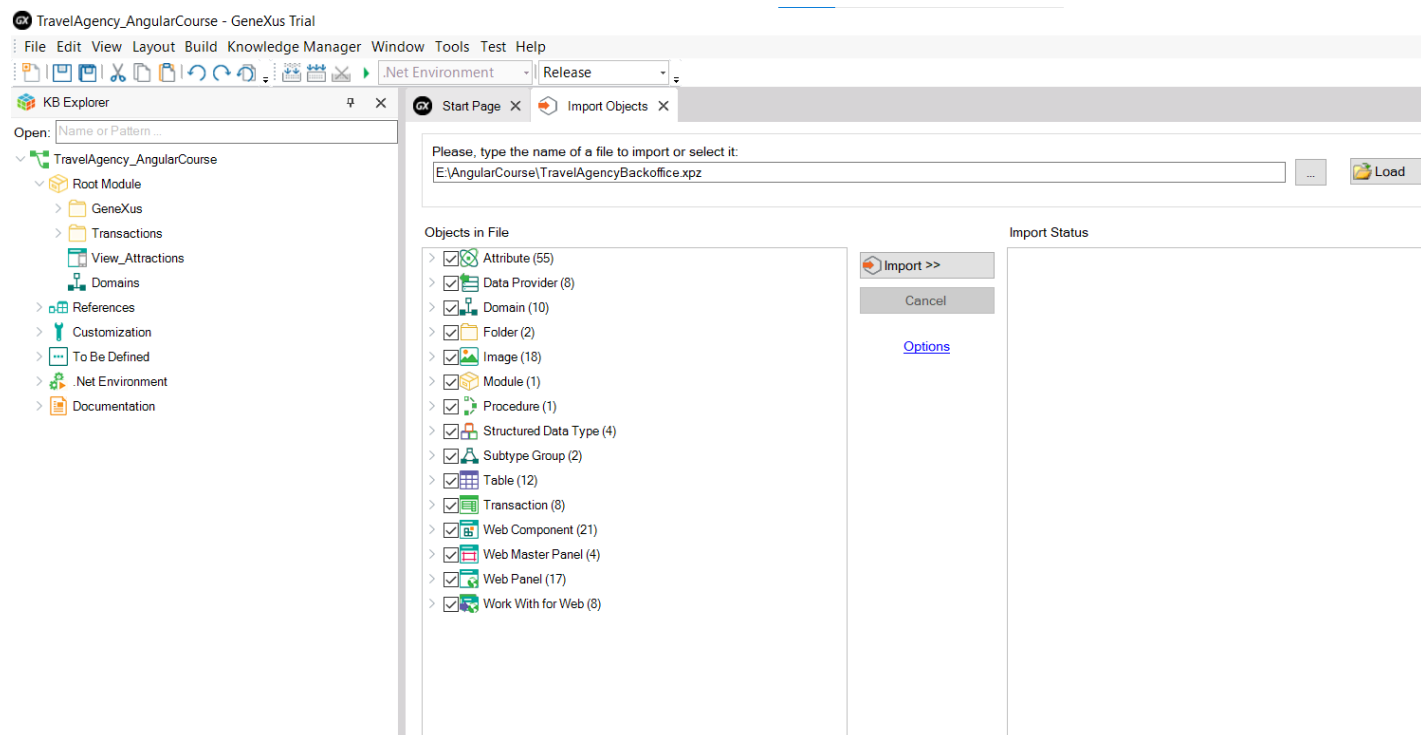
Se selecionamos endereços, vemos a rua onde está o museu do Louvre, sua caixa postal, cidade e país. Também vemos as linhas e paradas do metrô e as linhas de ônibus para acessar o museu.

Mais abaixo, vemos em um mapa a localização precisa da entrada.



Se pressionamos o botão da barra superior que diz Attractions, voltamos para a lista de atrações. E se pressionamos home, voltamos à tela inicial.

Nos vídeos a seguir veremos como completaremos estas funcionalidades, a princípio sem levar muito em conta questões da aparência como cores, formas de botões e estilo de fontes, já que nos concentraremos no desenvolvimento do que deve fazer a aplicação e de sua lógica, e uma vez feito isto, veremos como importar para nossa aplicação as definições de estética, realizadas por um profissional de design que nos ajudará a alcançar a aparência desejada.



Voltamos ao GeneXus e começamos a desenvolver a aplicação que vimos. A primeira coisa que fazemos é criar uma KB chamada `TravelAgency_AngularCourse`.

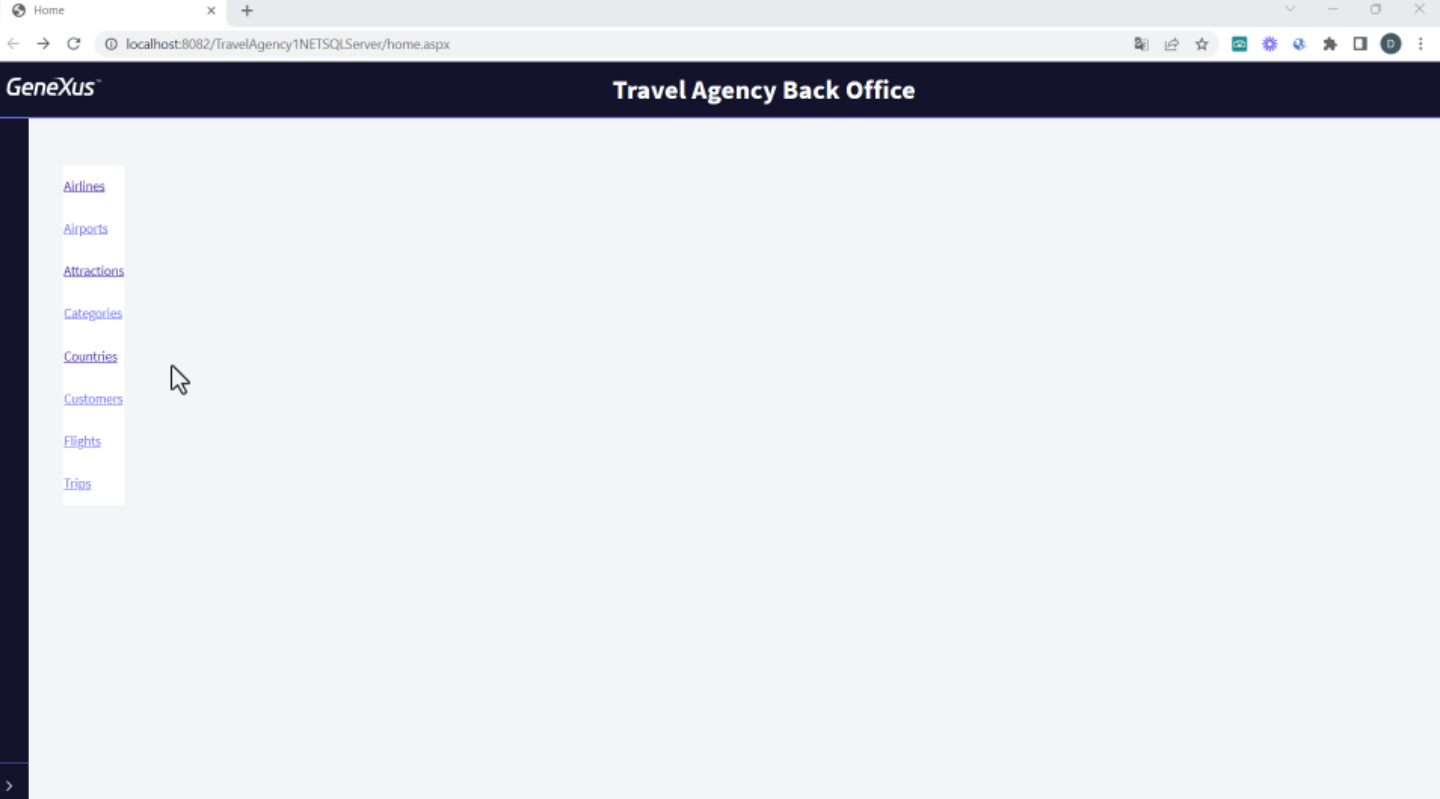
Agora importamos o modelo de dados já pronto, com todas as transações que utilizaremos, às quais já aplicamos o padrão `WorkWith for Web`, para que tenhamos disponível o backoffice da aplicação da agência de viagens.

Este backoffice web já está feito antes e nós o importamos para utilizá-lo como está, já que sua implementação não é o objetivo do curso. O que vamos desenvolver durante o curso é o front-end web voltado para customer-facing, em Angular.

Vamos para Knowledge Manager/Import e selecionamos o arquivo `TravelAgencyBackoffice.xpz`

Este xpz contém também data providers de carga de dados, portanto, ao executarmos será criada a base de dados, com suas tabelas já com dados.

Primeiro selecionamos o web panel Home como startup object e agora pressionamos F5 para nos familiarizar com as entidades da agência de viagens enquanto executamos o backoffice da aplicação.













Neste web panel Home, vemos um link para cada entidade. Se formos aos países, vemos que estão carregados vários países, cada um com suas cidades. Se formos para as atrações turísticas, vemos que atração pertence a um país e a uma cidade e tem informações detalhadas da atração.

Se voltamos ao Home, podemos navegar pelos demais dados da agência de viagens. Se tem dúvidas sobre os objetos transação, o padrão work with ou os data providers, sugerimos rever estes conceitos no curso GeneXus Core.

Attractions

INSERT

Search Name

Name	Country Name	Category Name	Photo	City Name	UPDATE	DELETE
Christ the Redemmer	Brazil	Monument		Rio de Janeiro	UPDATE	DELETE
Cinque Terre	Italy	Tourist site		Maranola	UPDATE	DELETE
Eiffel Tower	France	Monument		Paris	UPDATE	DELETE
Forbidden city	China	Tourist site		Beijing	UPDATE	DELETE
Glenfinnan Viaduct	Scotland	Tourist site		Glenfinnan	UPDATE	DELETE
London Towers	England	Monument		London	UPDATE	DELETE
Long Bridge	United States	Tourist site		San Francisco	UPDATE	DELETE
Louvre	France	Museum		Paris	UPDATE	DELETE
Matisse Museum	France	Museum		Nice	UPDATE	DELETE
Meet the Emperor	China	Monument		Beijing	UPDATE	DELETE

< >

Se formos para as atrações turísticas, vemos que atração pertence a um país e a uma cidade e tem informações detalhadas da atração.

Se voltamos ao Home, podemos navegar pelos demais dados da agência de viagens. Se tem dúvidas sobre os objetos transação, o padrão work with ou os data providers, sugerimos rever estes conceitos no curso GeneXus Core.

The screenshot displays the GeneXus development environment. On the left, the 'View_Attractions' panel is shown with a grid containing five columns: 'AttractionId', 'AttractionName', 'AttractionPhoto', 'CityName', and 'CountryName'. The 'MainTable' is also visible. In the center, the 'Properties' window for the 'Panel: View_Attractions' is open, showing various attributes such as Name, Description, Module/Folder, Qualified Name, Object Visibility, Main program, Master Panel, Caption, Miscellaneous, Network, and Main object properties. On the right, the 'KB Explorer' shows the project structure for 'TravelAgency_AngularCourse', with 'Front end' selected under the '.Net Environment'. The 'Properties' window for the 'Generator: Frontend (Front end)' is also open, showing configuration options like Name, Generate Android, Generate Apple, Main Platform, Dynamic Services URL, Services URL, SSL Pinning Pin Set, Smart Devices Cache, Generate Angular, Angular Specific, Setup Command, Run Command, Build Mode, and Default Platform.

Panel: View_Attractions	
Name	View_Attractions
Description	View_Attractions
Module/Folder	Root Module
Qualified Name	View_Attractions
Object Visibility	Public
Main program	True
Master Panel	(none)
Caption	View_Attractions
Miscellaneous	
Network	
Connectivity Support	Online
Deep Link Name	
Main object properties	
Application Title	

Generator: Frontend (Front end)	
Name	Frontend
Generate Android	False
Generate Apple	False
Main Platform	Angular
Dynamic Services URL	False
Services URL	https://trialapp
SSL Pinning Pin Set	
Smart Devices Cache	On
Generate Angular	True
Angular Specific	
Setup Command	npm install
Run Command	npm start
Build Mode	Prototype
Default Platform	Best fit

Voltamos ao GeneXus e começamos com a implementação da primeira tela da aplicação.

A agência de viagens nos solicita construir um painel que mostre uma lista horizontal das atrações disponíveis e quando selecionamos uma, ver os detalhes da atração.

Para isso, criamos um objeto Panel, que chamamos de View_Attractions.

Arrastamos para o form um grid e selecionamos os atributos AttractionId, AttractionName, AttractionPhoto, CityName e CountryName.

Vamos para as propriedades do panel e o definimos como objeto main.

Agora no Knowledgebase Navigator clicamos duas vezes no environment .NET, selecionamos Front end e vamos para suas propriedades. Definimos Generate Angular em True e, em seguida, Generate Android e Generate Apple em False. Observamos que a propriedade Main Platform ficou com o valor Angular.

The image shows two overlapping screenshots. The top screenshot is from the Node.js website, displaying the download page for Windows (x64). It features two green buttons: '18.17.1 LTS Recommended For Most Users' and '20.6.1 Current Latest Features'. Below these are links for 'Other Downloads | Changelog | API Docs'. A blue arrow points from the '18.17.1 LTS' button to the 'nodejs.org' link in the bottom screenshot.

The bottom screenshot is from the GeneXus Community Wiki, showing the page 'Angular Generator prerequisites for development environment'. The page title is '<Angular Generator prerequisites for development environment'. It states 'This documentation is valid for: GeneXus Beta'. Below, it says 'For Angular applications development, your development environment should include the following:'. A bullet point lists 'Node.js' with the instruction: 'Get Node.js from nodejs.org (download the LTS version). After the installation of Node.js, to check your version run the following command in a terminal/console window:'. A code block shows 'node -v'. Below the code block, it says 'Version has to be v12.17 or higher'. A sidebar on the left lists 'ANGULAR APPLICATIONS DEVELOPMENT' with sub-items: 'Getting started', 'Angular Overview', 'Requirements & execution', 'Prerequisites', 'Modeling', 'Roadmap', 'Troubleshooting', and 'Media'.

Antes de gerar nossa aplicação em Angular, devemos instalar algum software. Podemos ver isto se formos à wiki e procurarmos: “Angular requirements”.

Vemos que temos que instalar o Node.js a partir da página nodejs.org. Aqui, baixamos e instalamos a versão LTS. Já fiz isso em minha máquina.

For [Angular Applications Development](#), your development environment should include the following:

- **Node.js**

Get Node.js from nodejs.org (download the LTS version). After the installation of Node.js, to check your version run the following command in a terminal/console window:

```
node -v
```

- **npm package manager**

For the environment setup, npm packages will be installed (automatically) using the [npm client](#) command line interface, so you must have an npm package manager. Node.js includes it by default. So, after installing Node.js, check that you have the npm client installed, by running the following command in a terminal/console window (npm version should be 7 or higher):

```
npm -v
```

Uma vez instalada, para verificar a versão, abrimos uma janela de comando e escrevemos: **node -v**

Em seguida, temos que instalar um cliente gerenciador de pacotes npm, mas como já temos instalado o Node.js, ele já o inclui.

Para verificar se foi corretamente instalado, escrevemos “npm -v” e obtemos a versão.

Agora sim, temos tudo pronto para gerar e executar nossa aplicação em Angular.

```

Administrator: Windows PowerShell
.69 kB
node_modules_genexus_web-controls-library_dist_esm_gx-map-line_entry_js.js | - | 2
.61 kB
node_modules_genexus_web-controls-library_dist_esm_gx-gauge-range_entry_js.js | - | 2
.30 kB
node_modules_genexus_web-controls-library_dist_esm_gx-query-viewer-parameter_entry_js.js | - | 1
.63 kB
node_modules_genexus_web-controls-library_dist_esm_gx-audio_entry_js.js | - | 1
.59 kB
node_modules_genexus_web-controls-library_dist_esm_gx-query-viewer-element-format_entry_js.js | - | 1
.46 kB
node_modules_genexus_web-controls-library_dist_esm_gx-query-viewer-format-style_entry_js.js | - | 1
.45 kB
node_modules_genexus_web-controls-library_dist_esm_gx-query-viewer-element_entry_js.js | - | 1
.41 kB

Build at: 2021-08-03T13:15:59.371Z - Hash: 9d0ed77ac7a19ddd9fa8 - Time: 17217ms

Warning: E:\Models\TravelAgency_AngularCourse\CSharpModel\mobile\Angular\View_Attractions\node_modules\@genexus\web-standard-functions\dist\lib-esm\types\guid.js depends on 'uuid'. CommonJS or AMD dependencies can cause optimization bailout
s.
For more info see: https://angular.io/guide/build#configuring-commonjs-dependencies

** Angular Live Development Server is listening on localhost:54960, open your browser on http://localhost:54960/ **

√ Compiled successfully.
- Generating browser application bundles...

```














Voltamos ao GeneXus, vamos ao panel View_Attractions, clicamos com o botão direito e selecionamos Run.

Vemos que se abre uma janela de comandos e que são executadas várias linhas escritas pelo gerador Angular.

Ao terminar, é aberto automaticamente o navegador web e vemos a lista de atrações turísticas.

View_Attractions

localhost:64681/View_Attractions-Level_Detail

View_Attractions						
1	Eiffel Tower		France	Paris		
2	Glenfinnan Viaduct		Scotland	Glenfinnan		
3	Meet the Emperor		China	Beijing		
4	Christ the Redemmer		Brazil	Rio de Janeiro		
5	Rifugio Nuvolau		Italy	Belluno		
6	London Towers		England	London		
7	Louvre		France	Paris		
8	Forbidden city		China	Beijing		
9	Cinque Terre		Italy	Maranola		
10	Smithsonian Institute		United States	Washington		
11	Matisse Museum		France	Nice		
12	Long Bridge		United States	San Francisco		
13	The Great Wall		China	Beijing		

Já temos nossa primeira aplicação Angular funcionando.

Se observamos a URL, vemos que foi executada em localhost, dois pontos e uma porta atribuída dinamicamente. Isto corresponde ao servidor web instanciado para executar a aplicação Angular. A aplicação executada é View_Attractions-Level_Detail, que corresponde ao grid de atrações que incluímos no panel.

The screenshot shows the GeneXus IDE interface. On the left, a navigation tree displays 'View_Attractions' and 'Level_Detail_Grid1'. The main area shows a report titled 'Data Provider View_Attractions_Level_Detail_Grid1 Navigation Report'. The report details include:

- Name:** View_Attractions_Level_Detail_Grid1
- Description:** View_Attractions_Level_Detail_Grid1
- Status:** No Generation Required
- Output Devices:** None
- Environment:** C# Default (C#)
- Spec. Version:** 17_0_4-151606
- Form Class:** HTML
- Program Name:** View_Attractions_Level_Detail_Grid1
- Parameters:** in: &start, in: &count, in: &gxid, out: View_Attractions_Level_Detail_Grid1Sdt

Below the report details, the 'LEVELS' section is expanded to show a 'For Each Attraction' (Line: 2) loop. The loop configuration includes:

- Order:** AttractionId
- Index:** IATTRACTION
- Navigation filters:** Start from: FirstRecord, Loop while: NotEndOfTable
- Join location:** Server
- Optimizations:** Server Paging

The loop body contains three data provider calls:

- Attraction (AttractionId) INTO AttractionPhoto.Uri
- AttractionId AttractionName AttractionPhoto CountryId CityId
- Country (CountryId) INTO CountryName
- CountryCity (CountryId, CityId) INTO CityName

Se voltamos ao GeneXus e vemos a lista de navegação, vemos que o panel View_Attractions contém um nó chamado Level_Detail_Grid1 e que tem o símbolo de um data provider.

Se clicamos sobre ele, vemos que é a lista de navegação do data provider View_Attractions_Level_Detail_Grid1 que é o responsável por acessar a base de dados e recuperar as informações das atrações turísticas.

Se vemos onde diz Levels, encontramos um For Each Attraction já que como nosso grid tinha atributos, GeneXus encontrou uma tabela base e criou um for each implícito para acessar os dados da base de dados.

Vemos que efetivamente foi navegada a tabela Attraction para trazer os dados da atração e, em seguida, foi acessada a tabela Country para recuperar o nome do país e a tabela CountryCity para obter o nome da cidade.

Mais adiante veremos o que leva em consideração GeneXus para determinar a tabela base do grid e como se carrega a informação através do data provider cuja lista de navegação estamos vendo.

Disco1T (E:) > Models > TravelAgency_AngularCourse > CSharpModel > mobile > Angular > View_Attractions > src >

Name	Date modified	Type	Size
app	2/8/2021 14:26	File folder	
assets	2/8/2021 14:26	File folder	
environments	2/8/2021 14:26	File folder	
images	2/8/2021 14:26	File folder	
sass	2/8/2021 14:26	File folder	
translations	2/8/2021 14:26	File folder	
index.html	2/8/2021 14:26	File folder	
main.ts	2/8/2021 14:26	TypeScript Source File	
manifest.webmanifest	2/8/2021 14:26	File folder	
polyfills.ts	16/6/2021 14:31	TypeScript Source File	
test.ts	16/6/2021 14:31	TypeScript Source File	
tsconfig.app.json	16/6/2021 14:31	File folder	
tsconfig.spec.json	16/6/2021 14:31	File folder	

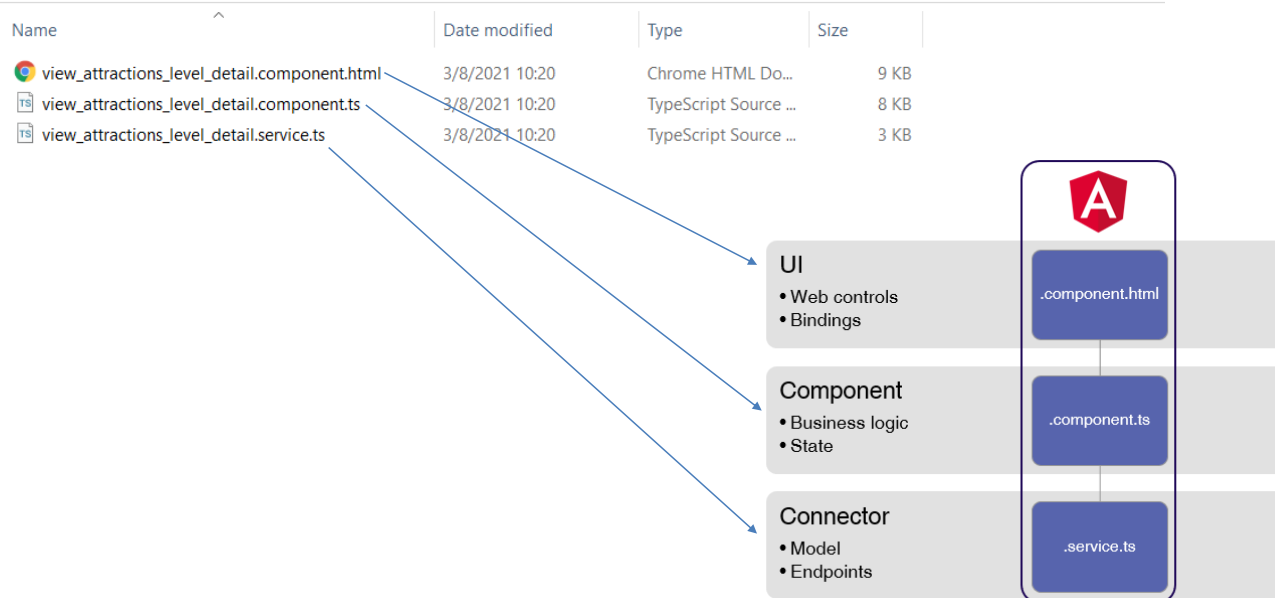
Name	Date modified	Type	Size
gx	2/8/2021 14:26	File folder	
View_Attractions	2/8/2021 14:26	File folder	
app.component.html	2/8/2021 14:26	Chrome HTML Document	4 KB
app.component.ts	2/8/2021 14:26	TypeScript Source File	12 KB
app.module.ts	2/8/2021 14:26	TypeScript Source File	3 KB
app.routing.ts	2/8/2021 14:26	TypeScript Source File	1 KB
app.settings.ts	2/8/2021 14:26	TypeScript Source File	3 KB
app-home.component.ts	2/8/2021 14:26	TypeScript Source File	1 KB
common.module.ts	16/6/2021 14:31	TypeScript Source File	4 KB
main.module.ts	2/8/2021 14:26	TypeScript Source File	1 KB
shared.module.ts	2/8/2021 14:26	TypeScript Source File	1 KB
shared-routing.module.ts	2/8/2021 14:26	TypeScript Source File	1 KB
styles.css	16/6/2021 14:31	Cascading Style Sheet	0 KB

Vamos analisar agora o que gera GeneXus, quando é executado o gerador Angular.

Se vamos para Tools/Explore Target Environment e depois vamos para mobile/Angular, encontramos uma pasta View_Attractions. Isto ocorre porque é o objeto main que executamos. Se agora selecionamos src/app, vemos as partes que compõem a aplicação.

Se clicamos em View_Attractions, vemos as partes que incluem o componente Angular, que são 3 arquivos.














Disco1T (E:) > Models > TravelAgency_AngularCourse > CSharpModel > mobile > Angular > View_Attractions > src > app > View_Attractions



O arquivo que termina em **component.html** contém a definição da interface de usuário da aplicação e é um template de Angular que representa o layout do panel. Aqui também estão incluídos elementos que Angular interpreta para associar os controles em tela com os dados da base de dados.

Então temos o arquivo que termina em **component.ts**, que é um arquivo TypeScript onde está definida a lógica do panel que implementamos. É aqui onde estão definidos os eventos e o estado dos controles da tela.

Por último, temos o arquivo que termina em **service.ts** que é responsável por interagir com os serviços do servidor web. Aqui ocorrem as comunicações com os serviços REST, são obtidos os dados da base de dados e são geradas as estruturas de dados que o panel vai utilizar.

View_Attractions				
1	Eiffel Tower		France	Paris
2	Glenfinnan Viaduct		Scotland	Glenfinnan
3	Meet the Emperor		China	Beijing
4	Christ the Redemmer		Brazil	Rio de Janeiro
5	Rifugio Nuvolau		Italy	Belluno
6	London Towers		England	London
7	Louvre		France	Paris
8	Forbidden city		China	Beijing
9	Cinque Terre		Italy	Maranola
10	Smithsonian Institute		United States	Washington
11	Matisse Museum		France	Nice
12	Long Bridge		United States	San Francisco
13	The Great Wall		China	Beijing



Se voltamos para a tela de atrações, vemos um grid que mostra os dados das atrações para baixo, que é o modo padrão.

No desenvolvimento do curso veremos como alterar esta visualização para poder paginar para a direita, bem como outros controles de tela que nos permitirão construir a aplicação com uma interface de usuário mais atrativa.

GeneXus™

training.genexus.com

wiki.genexus.com

training.genexus.com/certifications