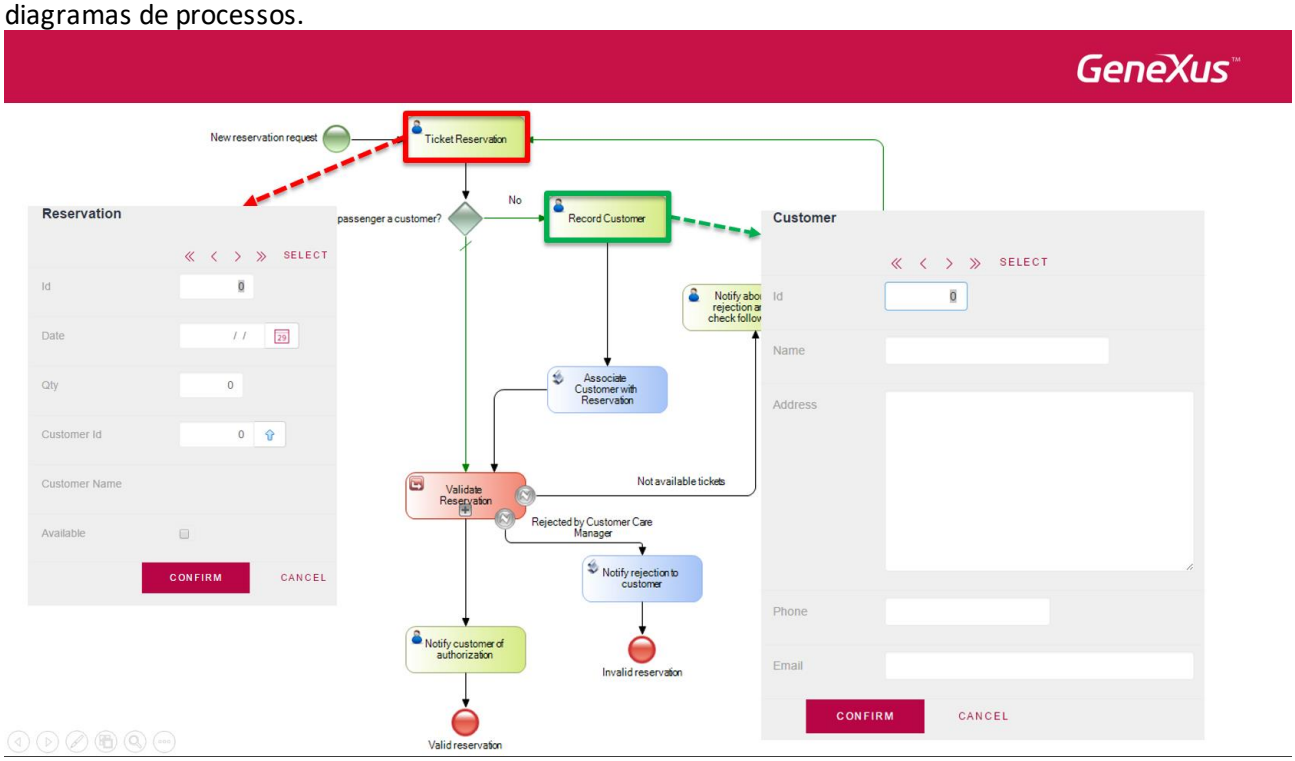
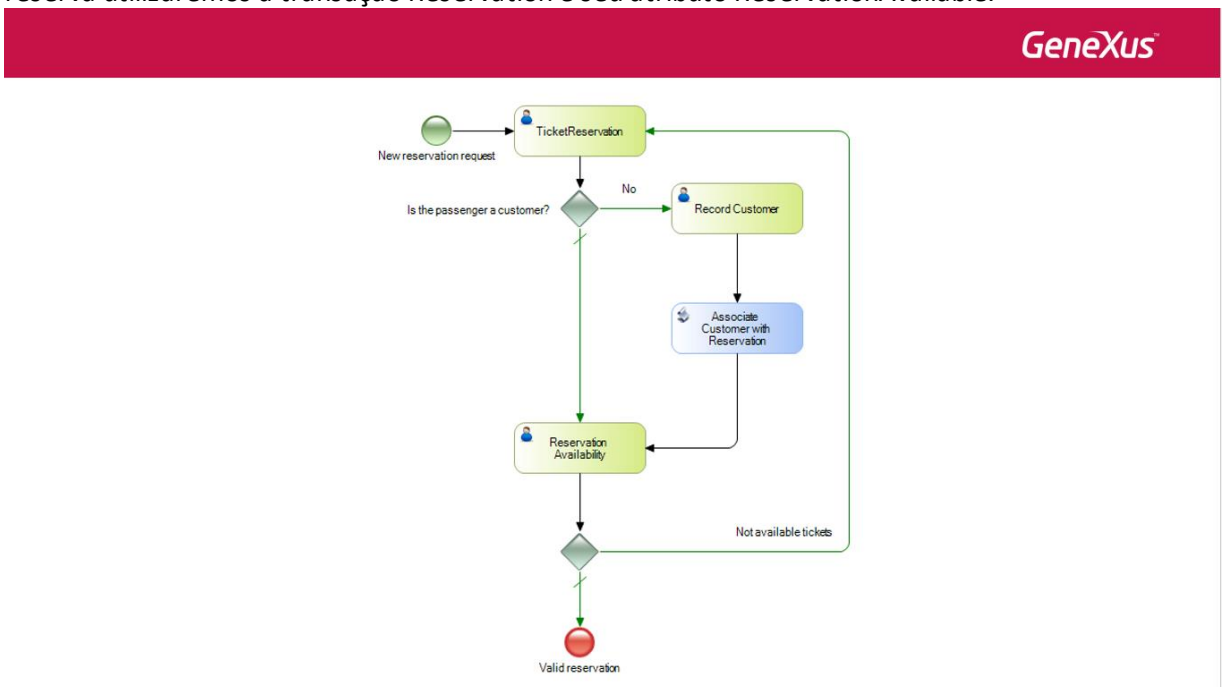
Executando um processo de negócios em um dispositivo móvel

Até agora temos associado objetos GeneXus que executavam em uma página web, as tarefas interativas dos diagramas de processos.



Mas também é possível associar as tarefas de usuário, objetos GeneXus que executem em um Smart Device. Por exemplo, suponhamos que o processo de adicionar uma reserva na agência de viagens, queremos executar desde um dispositivo móvel.

Vamos simplificar o processo de reserva de passagens que havíamos desenhado. Removemos o subprocesso de validação e alguns símbolos que não usaremos. Para especificar a disponibilidade da reserva utilizaremos a transação Reservation e seu atributo ReservationAvailable.

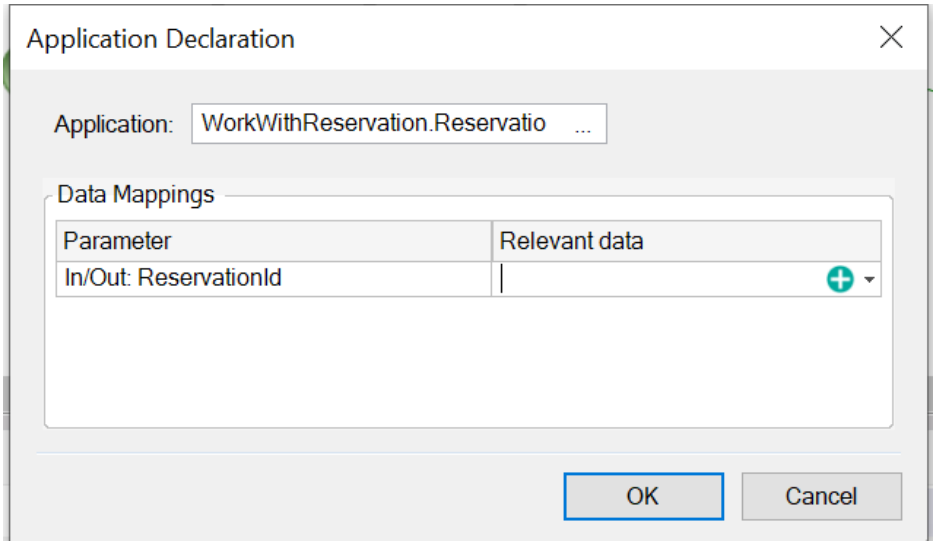


Logo selecionaremos a tarefa TicketReservation e para este exemplo, na propriedade Roles, excluímos o valor e o deixamos vazio. O mesmo fazemos com o None Start Event.

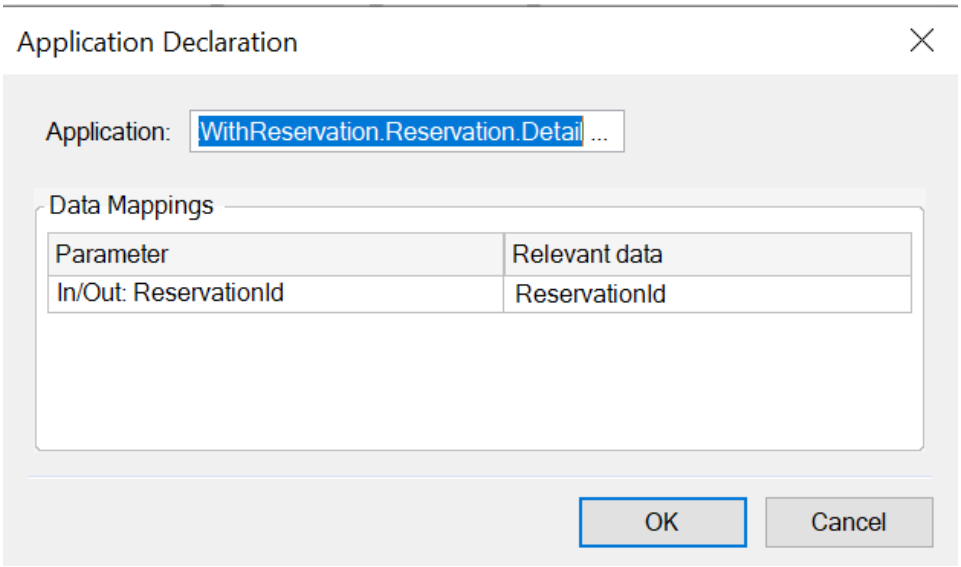
O primeiro que vamos fazer agora é aplicar o padrão Work With as transações Reservation e Customer, que são os envolvidos no processo. Para isso, , selecionamos ambas transações, clicamos com o botão direito, escolhemos Apply Pattern e Work With.

Vemos que abaixo do nó a transação Customer aparece o objeto SD: WorkWithCustomer e abaixo a transação Reservation, o objeto SD: WorkWitheservation.

Agora associaremos a tarefa TicketReservation, a aplicação SD que gerou o padrão. Para isso vamos as propriedades da tarefa e na propriedade Object pressionamos o botão e escolhemos WorkWithReservation.



Pressionamos a guia e associamos o dado relevante ReservationId.

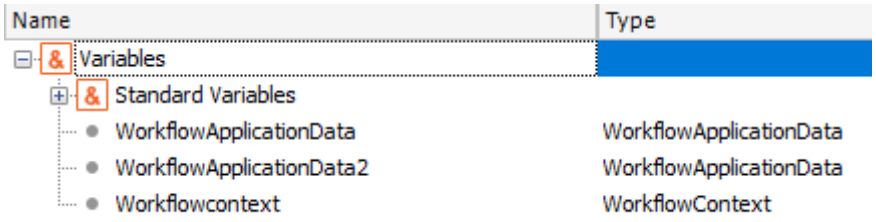


Pressionamos OK.
Algo muito importante a levar em conta quando associamos os objetos Smart Devices, **é que os dados relevantes não se mapeiam automaticamente entre o diagrama e o objeto SD**. Para fazer esta associação devemos contar com um procedimento.

Abrimos o objeto procedimento *ReservationMapRelevantData* que criamos previamente.
Na seção regras vemos que definimos uma regra Parm que recebe os identificadores de reserva e cliente, nos atributos ReservationId e CustomerId respectivamente.

```
1  Parm(in:ReservationId, in:CustomerId);
```

También definimos as seguintes variáveis:



E no source, utilizando a API de Workflow, obtemos o dado relevante ReservationId por seu nome e o associamos ao atributo ReservationId que recebemos por parâmetro.
O mesmo fazemos com o dado relevante CustomerId e o atributo CustomerId

```
1  &WorkflowApplicationData = &Workflowcontext.ProcessInstance.GetApplicationDataByName("ReservationId")
2  &WorkflowApplicationData.NumericValue = ReservationId
3
4  &WorkflowApplicationData2 = &Workflowcontext.ProcessInstance.GetApplicationDataByName("CustomerId")
5  if not CustomerId.IsNull()
6      &WorkflowApplicationData2.NumericValue = CustomerId
7  EndIf
8
9  Commit
```

Não devemos esquecer de incluir o Commit, quando trabalhamos com tipos de dados Workflow.

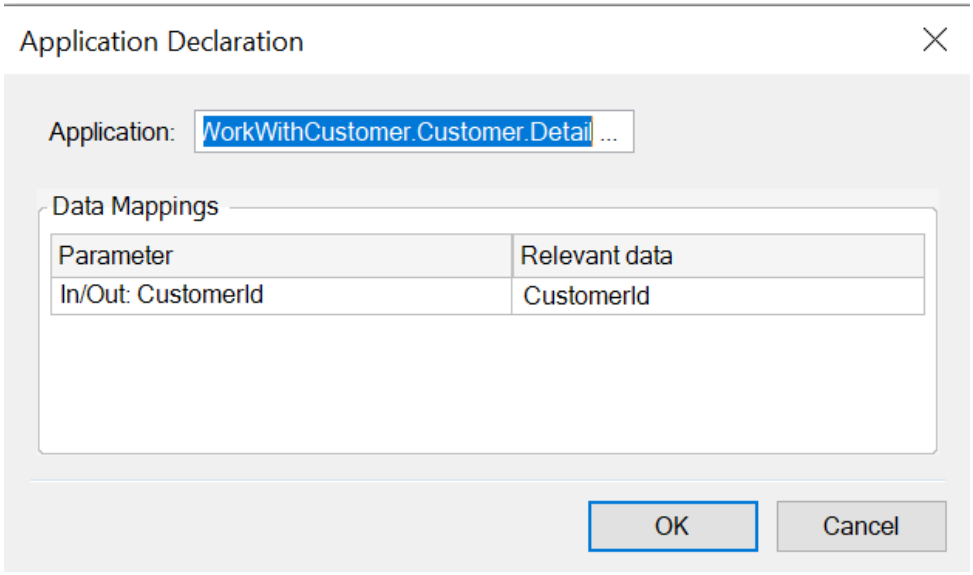
Como perguntamos pelo valor isNull() de CustomerId, vemos que adicionamos a seguinte regra a transação Reservation:

```
6  CustomerId.SetNull() If CustomerId.IsEmpty();
```

Agora abrirmos o objeto WorkWithsReservation, vamos a sua seção Detail/ Geral e modificamos o evento Save, adicionando a invocação ao procedimento que acabamos de criar.

```
Event 'Save'
    Composite
        SDActions.Save()
        ReservationMapRelevantData.Call(ReservationId,CustomerId)
        return
    EndComposite
EndEvent
```

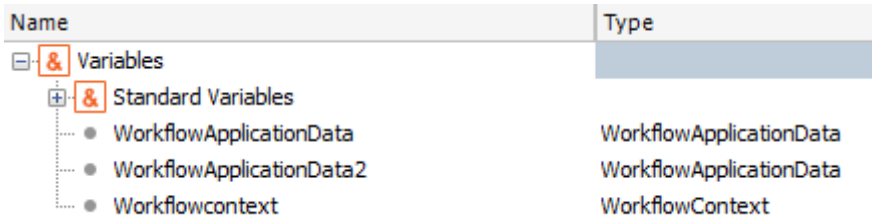
Desta maneira quando inserimos uma reserva nova desde a aplicação SD, foram mapeados os dados relevantes correspondentes.
Continuando, associamos a tarefa RecordCustomer a aplicação WorkWithCustomer e adicionamos o dado relevante CustomerId:



De forma similar ao que aconteceu com as reservas, temos que contar com um procedimento para associar o dado relevante CustomerId ao parâmetro CustomerId da transação Customer.
Para isso criamos previamente o procedimento de nome *CustomerMapRelevantData*. Vamos abri-lo.
Na seção de regras vemos a regra Parm que recebe o identificador do cliente no atributo CustomerId:

```
1 Parm(in:CustomerId);
```

Vemos as variáveis de tipos de dados Workflow definidas:



E no source implementado onde obtemos o dado relevante CustomerId por seu nome e o associamos ao atributo CustomerId que recebemos por parâmetro

```
1 &WorkflowApplicationData2 = &Workflowcontext.ProcessInstance.GetApplicationDataByName("CustomerId")
2 &WorkflowApplicationData2.NumericValue = CustomerId
3
4 Commit
```

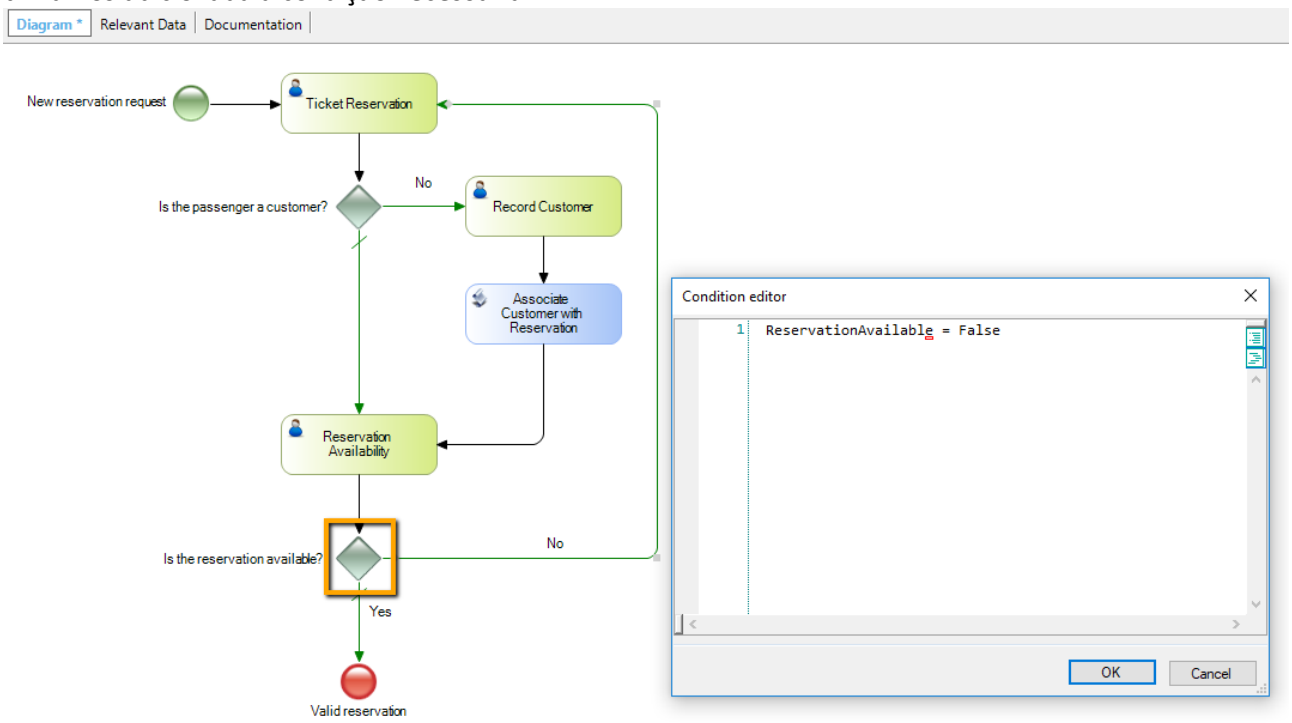
Agora vamos ao objeto WorkWithCustomer, na seção Detail / Geral e modificamos o evento Save, adicionando a invocação ao procedimento CustomerMapRelevantData.

```
Event 'Save'
    Composite
        SDActions.Save()
        CustomerMapRelevantData.Call(CustomerId)
        return
    EndComposite
EndEvent
```

Continuando com nosso diagrama, vamos adicionar a tarefa ReservationAvaikability ao objeto WorkWithReservations, de forma análoga a como fizemos antes com a tarefa Reservation.

Se a reserva está disponível termina o processo, mas se não está, devemos adicionar uma reserva nova, Esta avaliação fazemos com o exclusivo Gateway.

Se clicarmos duas vezes sobre o conector que une o Gateway com a tarefa TicketReservation, vemos que já tínhamos adicionada a condição necessária.



Para poder executar o diagrama de processos que construímos, devemos importar e configurar o **cliente GXflow para Smart Devices**. Aqui já temos feito, mas os detalhes encontram no seguinte link na tela:

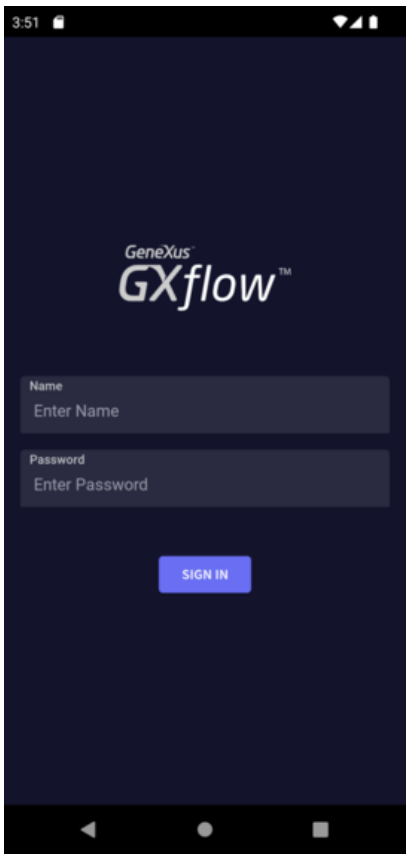
HowTo: Configuring GXflow Client For Smart Devices

Uma vez importado e configurado o cliente GXflow para SD, é necessário que tenhamos uma invocação a cada objeto SD usado em nosso diagrama de processos.
Para isso, devemos agregar o seguinte código ao dashboard WorkflowSDClient:

```
Event 'DummyCalls'
        WorkWithDevicesReservation.Reservation.Detail(1)
        WorkWithDevicesReservation.Reservation.List()
        WorkWithDevicesCustomer.Customer.Detail(1)
        WorkWithDevicesCustomer.Customer.List()
EndEvent
```

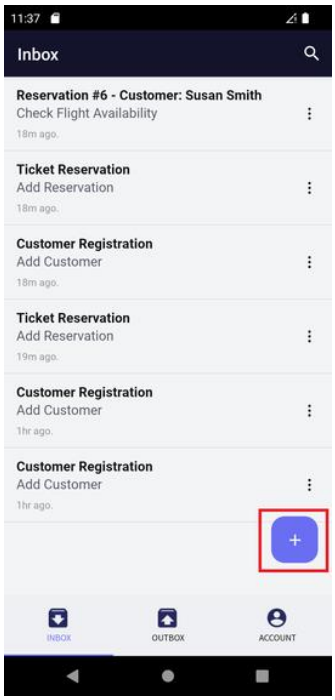
Agora estamos prontos para executar nosso processo de reserva de passagens em uma plataforma mobile.
Primeiro fazemos um Build All...
E agora pressionamos F5.

Vemos que o emulador de Android executou automaticamente mostrando a tela do login:

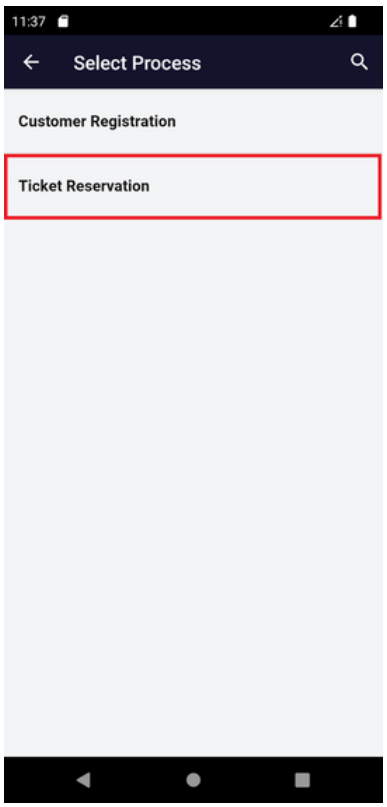


Vamos logar com o usuário administrador de workflow, assim que adicionamos como usuário: WFADMISTRATOR e o mesmo como password.
Ao entrar vemos uma tela que nos mostra a bandeja de entrada.

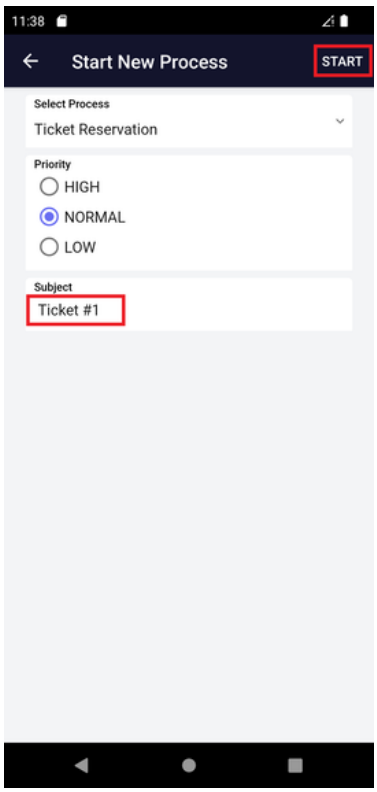
Selecionamos a bandeja de entrada e pressionamos o botão de '+' para instanciar um processo.



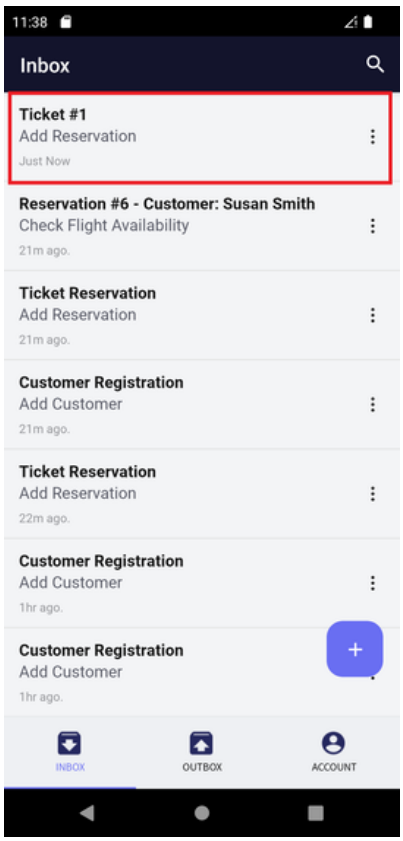
Agora selecionamos o processo FlightTicketReservation.



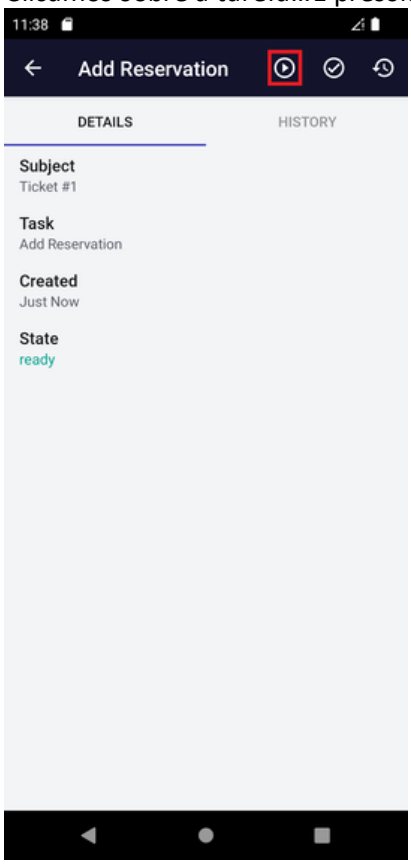
Vemos que abre a janela do processo. Pressionamos o botão de Start para iniciar o mesmo.



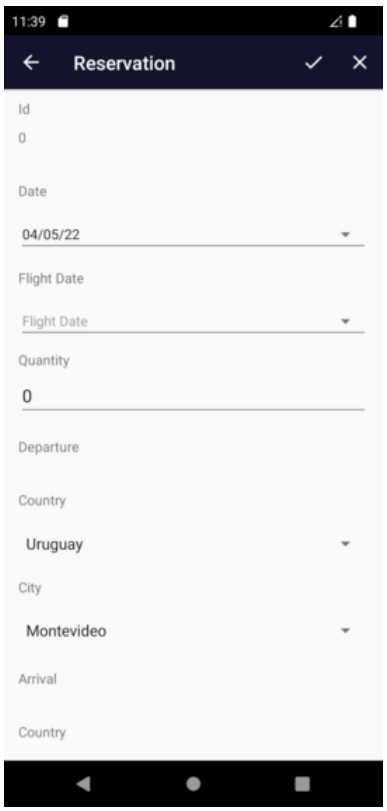
Agora vemos que o processo tem iniciado e que temos a tarefa TicketReservation pendente para executar.



Clicamos sobre la tarefa…E pressionamos o botão da data para iniciá-la.



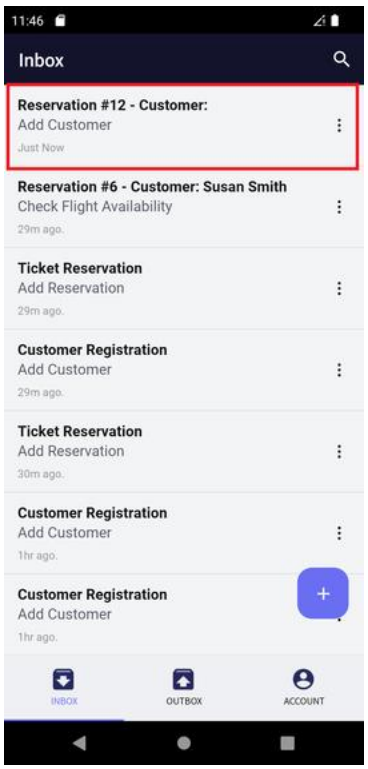
Vemos que abre o objeto SD para trabalhar com reservas, para que adicionemos uma reserva. Vamos adicionar uma reserva nova, deixamos o Id sem especificar já que é autonumerado e deixamos o identificador do cliente vazio.



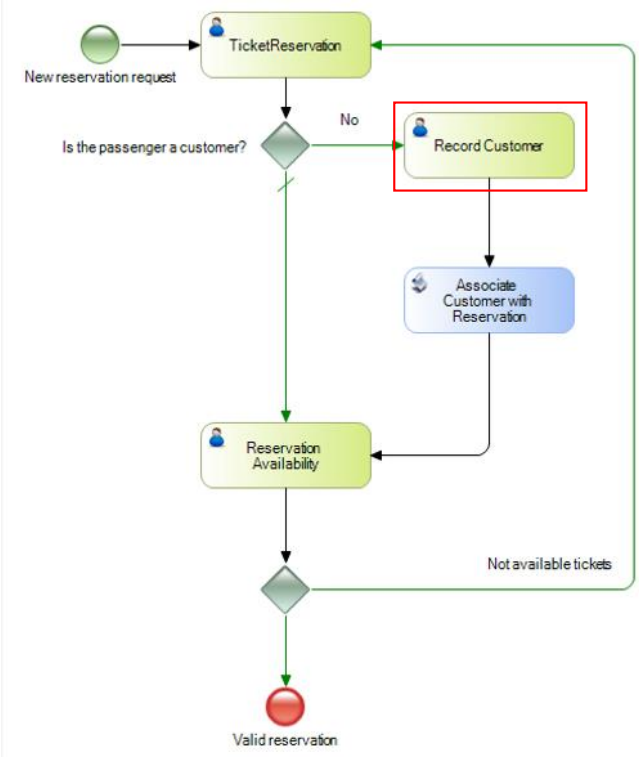
Agora pressionamos o botão de Confirmar.

Para finalizar a tarefa, escolhemos Completar.

Vemos que abre a bandeja de entrada e agora a tarefa pendente de execução é RecordCustomer.

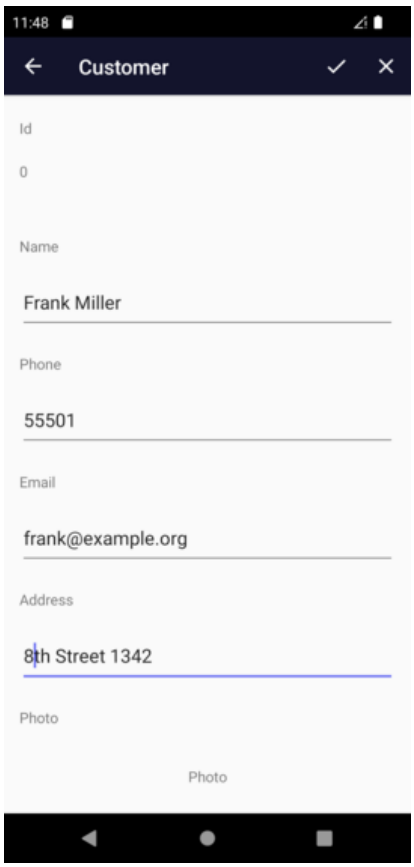


Como deixamos o cliente sem adicionar, o motor de workflow avaliou a condição do exclusive Gateway “Is the passenger a customer?” e determinou que a seguinte tarefa será RecordCustomer, a qual invocará o objeto SD Trabalhar com Clientes, para que adicionemos o cliente.



Executamos a tarefa Record Customer…

Adicionamos os dados do cliente e pressionamos Confirmar.

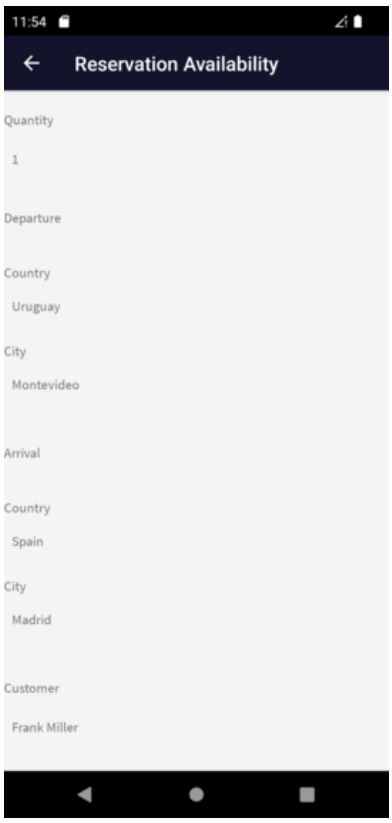


Para finalizar, completamos a tarefa Record Customer.

Agora a próxima tarefa que aparece como pendente é ReservationAvailability.

Clicamos sobre a mesma para abri-la

E a continuação a executamos. Observamos que o cliente foi com sucesso atributo a reserva, dado que executou corretamente o procedimento associado a tarefa batch AssociateCustomerToReservation.

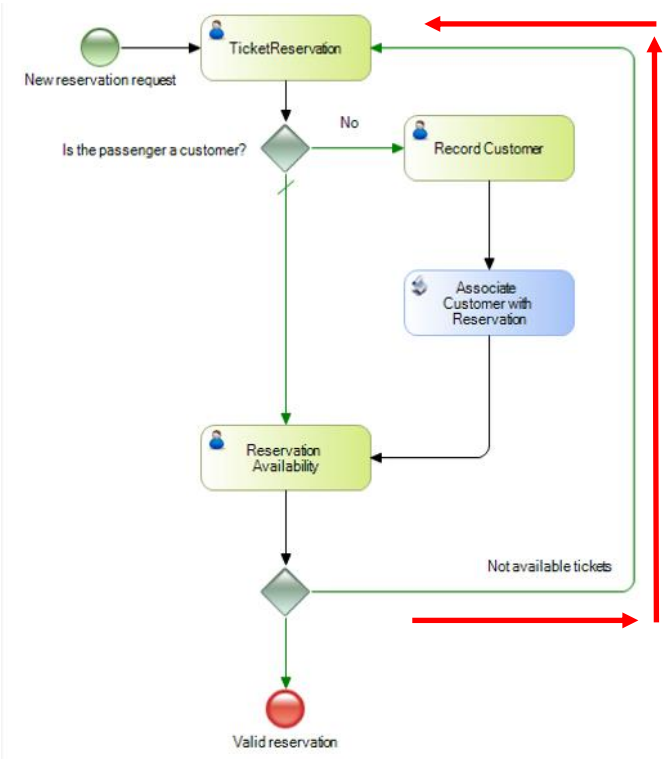


Vamos marcar que a reserva está disponível e pressionamos Confirmar.

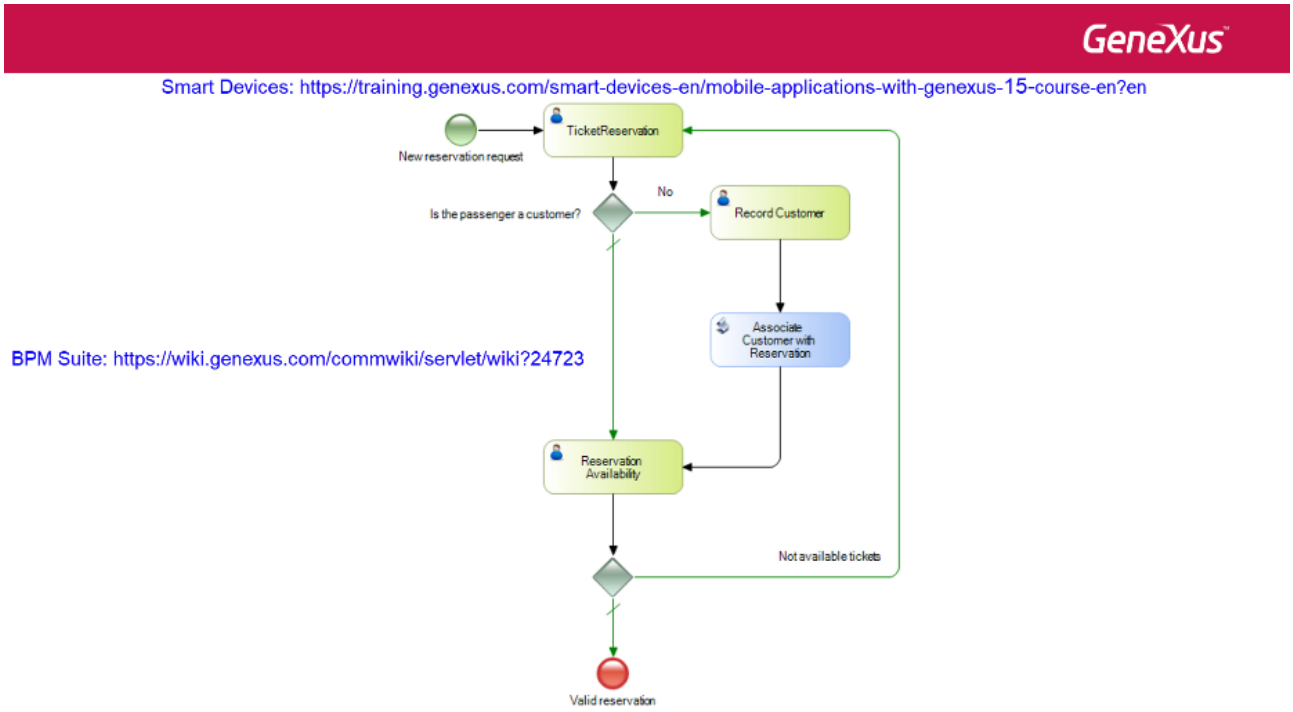
Para finalizar, completamos a tarefa ReservationAvailability.

Agora a bandeja de entrada não mostra mais tarefa pendentes o que implica que finalizou a execução do processo de reserva da agência de viagens.

Se tivéssemos marcado que a reserva não estava disponível, tivesse executado novamente o objeto SD WorkWithReservation, para que adicionarmos uma reserva nova.



Vimos assim como executar um diagrama de processo de negócios, no dispositivo móvel.
Em particular, temos associados as tarefas, os objetos gerados pelo padrão Work With, mais também poderíamos ter associado um objeto SD criado por nós, como por exemplo um panel para DS.
Neste caso temos gerado a aplicação para Smart Devices em Android e executamos a mesma utilizando um emulador, mas é possível prototipar sobre um dispositivo físico e gerar aplicações para outras plataformas, como por exemplo, dispositivos iOS como Ipad ou Iphone. Para saber mais sobre aplicações Smart Devices, visite o link que mostra na tela:



E para conhecer mais possibilidades o suite BPM de GeneXus visite o seguinte link do wiki.