

## Agendamento

### Transcrição

[00:00] Agora vamos implementar o método “Salvar” da classe “AgendamentoDAO” para inserir no banco de dados o novo registro de agendamento. Vamos acessar a conexão que já foi passada aqui para classe “AgendamentoDAO”, eu faço conexão e aqui eu vou inserir “.Insert” e eu vou passar aqui dentro a instância do agendamento que eu estou recebendo, “agendamento” e eu vou tentar agora rodar a aplicação para ver se funciona.

[00:37] Deu um monte de erros aqui, porque fizemos uma refatoração e não completamos, então vamos ter que parar um pouquinho, fazer essa refatoração para o nosso código poder compilar.

[00:49] Vamos aqui na ViewModel, no agendamento ViewModel e vai resolver esse erro aqui do veículo, vamos substituir a propriedade “Veículo” por duas propriedades, que são o modelo e o preço do veículo, vamos remover essa propriedade e colocar aqui uma propriedade chamada “modelo”, que é uma string, no “get” eu vou pegar “this.Agendamento.Nome” que vai ser o nome do veículo e aqui eu vou fazer “this.Agendamento.Nome = value”.

[01:33] Vou fazer a mesma coisa agora para o preço, vou fazer uma outra propriedade chamada preço, que vai ser um decimal, “preco” e aqui no retorno vou retornar “this.Agendamento.Preco” e “set” também vou colocar “this.Agendamento.Preco = value”, aqui ainda tem que corrigir esse modelo, esse “Nome” não é certo, tem que ser modelo, então “Agendamento.Modelo”.

[02:13] Agora estou com um problema porque o “set” está tentando modificar umas propriedades como privadas, vamos ter que modificar também, vamos ter que refatorar isso aqui e trocar para público.

[02:28] Então tira o private dessas propriedades, tirando aqui também, aqui do “dataAgendamento” também e do agendamento também. Tem mais alguns erros, vamos ver, está certo aqui, eu vou remover essa linha, “Agendamento.Veiculo” eu vou remover, porque não estamos usando mais veículo dentro do agendamento e aqui dentro do “new Agendamento” eu vou ter que passar alguns parâmetros, alguns parâmetros aqui como o nome, fone, email, vão vir de usuário e modelo e preço vão vir do veículo.

[03:10] Então “AgendamentoViewModel” também vai ter que receber um parâmetro que é o usuário, porque sem esse usuário eu não consigo preencher o agendamento aqui embaixo, eu vou receber esse parâmetro novo, “usuário” e aqui no “new Agendamento” eu passo nome do usuário, vou passar o fone, telefone, eu passo o e-mail e passa agora os dados do veículo que são “veiculo.modelo” que no caso é nome e “veiculo.preco”.

[03:54] Ainda tem mais alguns erros para corrigir aqui no AgendamentoViewModel, vamos ver, o “carro” está pegando “Veiculo.Nome”, mas eu quero que ele pegue diretamente a nova propriedade que é o modelo, que é o nome do veículo. Aqui o preço ele vai pegar diretamente da propriedade nova que é o preço.

[04:14] Ainda tem mais alguns erros, aqui no salvar eu estou pegando “Veiculo.Nome”, eu quero agora pegar o modelo diretamente. E aqui no “Veiculo.Preco” vou trocar para somente preço, agora vamos compilar de novo, ver se aparece mais algum erro.

[04:33] Como é de se esperar, como acrescentamos um parâmetro aqui no construtor de “AgendamentoViewModel”, lá no “AgendamentoViewModel” o “xaml.cs”, que é o Code behind, que criamos um novo Model e temos que passar aqui o usuário que vai ser preenchido aqui nessa instância nova do “AgendamentoViewModel”.

[04:55] Eu vou ter que passar aqui o usuário, só que o usuário ainda não existe aqui, o usuário não é conhecido ainda pelo View, não é conhecida pelo Code behind, vamos ter que passar também no construtor do “AgendamentoView”, uma nova instância “Usuário usuario” e vou passar aqui para o “AgendamentoViewModel”.

[05:23] Agora vou copiar de novo para ver se não tem nenhum erro, apareceu mais um erro, porque modificamos o construtor de mais uma classe, estamos propagando os erros, vamos ver, agora está dando erro em “DetalheView.xaml.cs”, que é o code behind do DetalheView. Aqui ele está passando no construtor de AgendamentoView, porque está passando o veículo que é esse “msg” que vem em uma mensagem, eu vou trocar o nome aqui para “veículo”, vou trocar aqui em cima, vou renomear, “veículo”.

[06:04] Agora eu vou ter que passar aqui dentro o usuário também, só que o usuário não é conhecido aqui em DetalheView, o que vamos fazer? Vamos criar aqui um novo campo nessa classe, já temos o “veiculo” e vamos criar um usuário.

[06:26] Criamos aqui uma propriedade nova, um DetalheView chamada “Usuario Usuario”, agora estamos com o set público, vamos mudar para privado, mudando para privado os dois e aqui no construtor DetalheView eu estou recebendo somente o veículo, eu tenho que receber veículo e também o usuário, eu vou passar para propriedade usuário, “this.Usuario = usuario”.

[07:04] Agora vamos ver esse erro que está dando aqui embaixo, aqui eu tenho que passar, não usuário em minúsculo mas sim a propriedade “this.Usuario”, vamos ver se compila agora, está dando um erro no code behind da ListagemView, porque ele está tentando instanciar uma DetalheView, só que ele está passando somente o veículo e não está passando usuário.

[07:28] Vamos passar aqui no lugar de “msg”, vamos mudar “msg” para veículo, agora veículo ele está passando e falta passar o usuário, agora para passar aqui para o DetalheView uma instância de usuário essa classe “ListagemView.xaml.cs” também precisa conhecer o usuário e ela não conhece, vamos ter que passar também no construtor da ListagemView uma instância de usuário.

[07:54] Eu crio um parâmetro novo aqui chamado usuário, “Usuario usuario” e vou passar para uma propriedade local essa Instância de usuário, como que eu faço?

[08:05] Eu crio aqui “Usuario usuario”, aqui no escopo da classe, só que esse usuário vai ser imutável, ele vai ser recebido uma vez e não vai mudar de novo, deixa eu mudar aqui o nome “usuário”, acertei, agora para deixar imutável eu coloco o quê?

[08:25] A palavra-chave “readonly” para ele ficar somente leitura, ela só vai ser atribuída uma vez aqui no construtor, “this.usuario = usuario”, com isso conseguimos guardar localmente uma Instância do usuário e esse o usuário vamos usar lá embaixo, quando instanciar o “DetalheView”, eu tenho o veículo e vou passar para o DetalheView também o usuário, agora compilamos para ver se tem mais algum erro.

[08:57] Não apareceu mais nenhum erro, vamos refaturar mais alguma coisa? Vamos definir algumas mensagens.

[09:06] Eu vou extrair esse trecho de código porque esse método “OnAppearing” está ficando muito grande. Eu vou clicar aqui com o botão direito, Quick Actions e vou extrair um método, eu vou chamar “AssinarMensagens” e aqui embaixo no “OnDisappearing” eu vou selecionar essas duas linhas e vou extrair também o outro método, que eu vou chamar de cancelar assinatura, “CancelarAssinatura”.

[09:40] Agora vamos rodar a aplicação para ver se está funcionando, eu vou passar aqui o usuário “joao@alura.com.br”, senha “alura123”, vou entrar agora, apareceu um erro aqui, construtor default não encontrado para o tipo ListagemView, modificamos a classe ListagemView para incluir um parâmetro no construtor, só que estamos

instanciando essa `ListagemView` em algum lugar e não está passando esse parâmetro usuário no construtor, vamos dar uma olhada.

[10:15] Consumimos onde a `ListagemView`? Consumimos na `MasterDetailView`, então vamos abrir a `MasterDetailView.xaml`, é o que está acontecendo aqui, estamos criando no detalhe do `MasterDetailPage` uma instância de um `“NavigationPage”` e dentro da `“NavigationPage”` estamos passando uma instância nova do `ListagemView`.

[10:43] Estamos passando em uma instância sem passar o parâmetro exigido no construtor, que é o usuário, ou adicionamos aqui usuário, aqui no XAML de alguma forma ou vamos fazer isso via código CSharp, eu prefiro fazer via CSharp porque é mais simples, muito mais simples, muito mais prático, eu vou remover essa `MasterDetailPage.Detail`, removendo aqui esse trecho de código e vou “settar” lá no Code behind diretamente no código CSharp, eu vou declarar o detalhe dessa `MasterDetail`.

[11:21] Aqui no construtor do `MasterDetailView` já estamos declarando, já estamos definindo que está aqui o `MasterView`, agora só falta declararmos também, definir qual é o detalhe que vai ser o nosso `“NavigationPage”`, `“this.Detail”` vai ter que ser um `“new NavigationPage”` que vai receber uma nova `“ListagemView”`, aqui dentro eu passo meu `“ListagemView”`, quando eu faço isso ele mostra para mim o construtor exigindo o usuário.

[12:02] Agora nessa parte, no nosso código CSharp eu consigo acessar o usuário, esse usuário eu vou passar lá para dentro da `ListagemView`, agora vamos rodar o código e ver se conseguimos sucesso dessa vez.

[12:14] Eu vou passar o usuário `“joao@alura.com.br”`, `“alura123”`, agora vamos entrar, ele foi sem problemas, foi para `ListagemView`, agora vamos para próxima página, fomos agora para página de detalhes, sem problemas também, agora vamos clicar no próximo para ir para página de agendamento. Conseguimos chegar na página de agendamento, onde já temos preenchidos todos os dados do João da Silva, que é o nosso usuário que foi autenticado e com isso não precisamos ficar mais digitando os dados do formulário do agendamento.