**Custom Master**-**details:**

<**CrudComponent** **TEntity**="Customer" **UpsertData**="customer" **OnUpsert**="async formData => await

UpdateDatabaseAsync(formData)">

<**DataGrid** @ref="grid" **Search**="search" **TableHeight**="400">

<**Columns**>

<**GridColumn** **Field**="t => t.CustomerType" />

<**GridColumn** **Field**='t => t.CustomerType == CustomerType.Real ? t.FName + " " + t.LName :

t.CompanyName' **Title**="Name" />

<**GridColumn** **Field**="t => t.MobileNumber.CallFormat(' ')" />

<**GridCommandColumns** />

</**Columns**>

</**DataGrid**>

<**Window** **OnOpen**="async () => await loadGroups()" @ref="window" **Title**="Customer & Groups"

**Style**="width:450px;">

<**Content**>

<**CaspianForm** **Model**="customer">

<**TabPanel**>

<**Tabs**>

<**TabPanelItem** **Title**="Customer">

<div class="row p-4 pt-2 pb-2 c-controls" style="height:320px">

<div class="col-md-4">

<label>Customer Type</label>

</div>

<div class="col-md-8">

<**DropdownList** @bind-Value="customer.CustomerType" />

</div>

<div class="col-md-4">

<label>First Name</label>

</div>

<div class="col-md-8">

<**StringTextBox** @bind-Value="customer.FName"

**Disabled**="customer.CustomerType == CustomerType.Legal" />

</div>

<div class="col-md-4">

<label>Last Name</label>

</div>

<div class="col-md-8">

<**StringTextBox** @bind-Value="customer.LName"

**Disabled**="customer.CustomerType == CustomerType.Legal" />

</div>

<div class="col-md-4">

<label>Gender</label>

</div>

<div class="col-md-8">

<**DropdownList** @bind-Value="customer.Gender"

**Disabled**="customer.CustomerType == CustomerType.Legal" />

</div>

<div class="col-md-4">

<label>Company Name</label>

</div>

<div class="col-md-8">

<**StringTextBox** @bind-Value="customer.CompanyName"

**Disabled**="customer.CustomerType == CustomerType.Real" />

</div>

<div class="col-md-4">

<label>Mobile Number</label>

</div>

<div class="col-md-8">

<**StringTextBox** @bind-Value="customer.MobileNumber" **MaxLength**="11" />

</div>

<div class="col-md-4">

<label>Tel</label>

</div>

<div class="col-md-8">

<**StringTextBox** @bind-Value="customer.Tel" />

</div>

</div>

</**TabPanelItem**>

<**TabPanelItem** **Title**="Groups">

<div class="p-4" style="height:320px">

<**CheckboxList** **TValue**="int" **TEntity**="CustomerGroup" @bind-Values="groupsId"

**TextExpression**="t => t.Title" />

</div>

</**TabPanelItem**>

</**Tabs**>

</**TabPanel**>

<div class="pt-4 pt-0 pb-2">

<**FormCommandButton** />

</div>

</**CaspianForm**>

</**Content**>

</**Window**>

</**CrudComponent**>

@code

{

Window window;

DataGrid<Customer> grid;

Customer customer = new Customer();

Customer search = new Customer();

IList<int> groupsId;

async Task loadGroups()

{

if (customer.Id > 0)

{

///In update we should fill groupsId field

using var scope = CreateScope();

groupsId = await new CustomerGroupMembershipService(scope.ServiceProvider).GetAll()

.Where(t => t.CustomerId == customer.Id).Select(t => t.CustomerGroupId).ToListAsync();

}

else

groupsId = new List<int>();

}

async Task UpdateDatabaseAsync(FormData<Customer> form)

{

form.Cancel = true;

using var scope = CreateScope();

var provider = scope.ServiceProvider;

Customer result = customer;

IList<CustomerGroupMembership> removedGroups = null;

var service = new CustomerService(provider);

///Only in customer update we should remove groups

if (customer.Id > 0)

{

///The groups removed from list and exist in database, this groups should removed from database

removedGroups = await new CustomerGroupMembershipService(provider)

.GetAll().Where(t => t.CustomerId == customer.Id && !groupsId.Contains(t.CustomerGroupId))

.ToListAsync();

}

IEnumerable<CustomerGroupMembership> addedGroups = null;

var transaction = await service.Context.Database.BeginTransactionAsync();

if (customer.Id == 0)

{

result = await service.AddAsync(customer);

///we should doing insert to fill Id of customer

await service.SaveChangesAsync();

///after customer id fill from database

addedGroups = groupsId.Select(t => new CustomerGroupMembership()

{

CustomerId = customer.Id,

CustomerGroupId = t

});

}

else

{

await service.UpdateAsync(customer);

var oldGroupsId = await new CustomerGroupMembershipService(provider).GetAll()

.Where(t => t.CustomerId == customer.Id).Select(t => t.CustomerGroupId).ToListAsync();

addedGroups = groupsId.Where(t => !oldGroupsId.Contains(t)).Select(t => new

CustomerGroupMembership()

{

CustomerGroupId = t,

CustomerId = customer.Id

});

}

var membershipService = new CustomerGroupMembershipService(provider);

/// Add groups

await membershipService.AddRangeAsync(addedGroups);

///Remove groups

membershipService.RemoveRange(removedGroups);

await service.SaveChangesAsync();

await transaction.CommitAsync();

///After transaction commited we can select row

await grid.SelectRowById(result.Id);

/// we must close window on last

await window.Close();

}

}