Code Document: Church Management System - Rock RMS

Worked on below list of pages.

Groups

Group members

Group

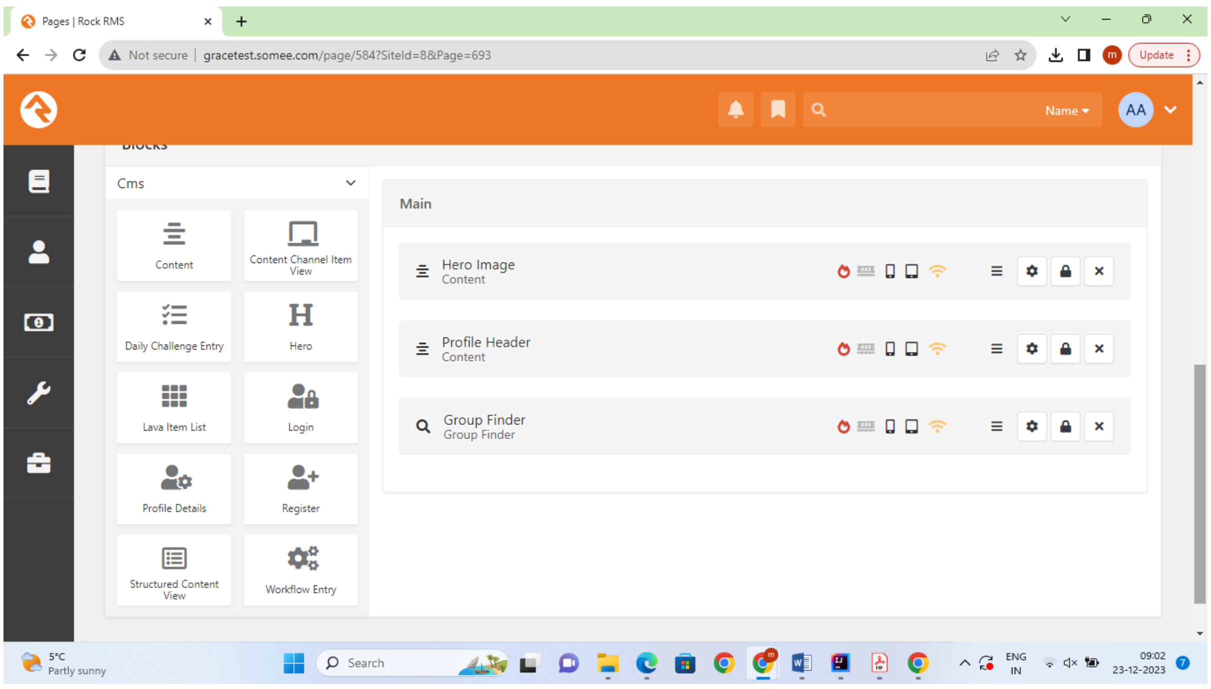
Teams

Team

Login

Chat

Groups:



The below code and explanation are of the block Hero Image Content block:

```
<StackLayout>

  <Grid>

    <Image Source="https://assets-global.website-
files.com/5f6b9a421d5a61e1d0cd9e3d/62f285c4f9aa3441840257d6_nathan-mullet-pmiW630yDPE-
unsplash.jpg"

        Aspect="AspectFill"  />



    <BoxView BackgroundColor="#80000000"/>
    <Label Text="Groups"
        TextColor="White"
        FontSize="36"
        FontAttributes="Bold"
        HorizontalOptions="Center"
        VerticalOptions="Center" />
  </Grid>
</StackLayout>
```

1. **<StackLayout>**: This is a layout container that stacks its child elements vertically by default. It's a common layout container in Xamarin.Forms.
2. **<Grid>**: This is a layout container that allows you to create a grid-based layout. It is used to arrange child elements in rows and columns.
3. **<Image>**: An **Image** element is used to display an image. In this case, it is displaying an image from the specified URL. The **Aspect="AspectFill"** property ensures that the image fills the available space while maintaining its aspect ratio.
4. **<BoxView BackgroundColor="#80000000"/>**: This is a **BoxView** element, which is a simple rectangular box that can be used for various purposes. In this case, it serves as an overlay with a semi-transparent black background (**#80000000** represents a dark translucent background color).
5. **<Label>**: This is a **Label** element displaying the text "Groups." It is styled with white text color, a font size of 36, and bold font attributes. The **HorizontalOptions** and **VerticalOptions** properties are set to "Center," ensuring that the label is centered within the grid.

The below code and explanation are of block “Profile Header”:

```
<StackLayout Margin="0,0,0,0" Orientation="Vertical" VerticalOptions="Center"
BackgroundColor="#7bb446">
  <StackLayout Margin="0,10,0,10" Orientation="Horizontal" HorizontalOptions="Start">
    <Rock:Icon IconClass="user" StyleClass="text-primary" Margin="10,0,0,0" TextColor="White"
VerticalOptions="CenterAndExpand" FontSize="24" />
    <Label Grid.Column="1" StyleClass="pl-3" Text="Hi {{ CurrentPerson.FirstName }}!"
TextColor="White" Margin="0,10,0,10" VerticalOptions="CenterAndExpand">
      <Label.GestureRecognizers>
        <TapGestureRecognizer Command="{Binding PushPage}" CommandParameter="09ee7ee8-
0b44-4b88-b5ca-4dc26dec006f" />
      </Label.GestureRecognizers>
    </Label>
  </StackLayout>


  <Rock:Divider />
</StackLayout>
```

1. **<StackLayout Margin="0,0,0,0" Orientation="Vertical" VerticalOptions="Center" BackgroundColor="#7bb446">**: This is the outermost **StackLayout** that serves as a container for the entire layout. It has a green background color (#7bb446) and is vertically centered on the screen.
2. **<StackLayout Margin="0,10,0,10" Orientation="Horizontal" HorizontalOptions="Start">**: This is a nested **StackLayout** within the outer one. It is oriented horizontally and has a margin of 10 units at the top and bottom. The **HorizontalOptions** is set to "Start," aligning its content to the left.
3. **<Rock:Icon IconClass="user" StyleClass="text-primary" Margin="10,0,0,0" TextColor="White" VerticalOptions="CenterAndExpand" FontSize="24" />**: This is a custom **Rock:Icon** control displaying a user icon. It has a white color, a margin to the left, and a font size of 24.
4. **<Label Grid.Column="1" StyleClass="pl-3" Text="Hi {{ CurrentPerson.FirstName }}!" TextColor="White" Margin="0,10,0,10" VerticalOptions="CenterAndExpand">**: This is a **Label** displaying a greeting message. It uses data binding (**{{ CurrentPerson.FirstName }}**) to dynamically display the user's first name. It has a white text color, a margin, and is vertically centered and expanded.
5. **<Label.GestureRecognizers>**: This block contains a **TapGestureRecognizer** that is associated with the **Label**. It binds to the "PushPage" command in the view model, passing a specific parameter (ID) when tapped.
6. **<Rock:Divider />**: This is a custom **Rock:Divider** control, which is likely a visual separator/divider element.

Overall, the code represents a user interface with a green background, a user icon, a greeting label with a dynamic first name, and a divider below it. The label has a tap gesture recognizer that triggers a command in the view model when tapped.

The below code and explanation are of block “Group Finder”

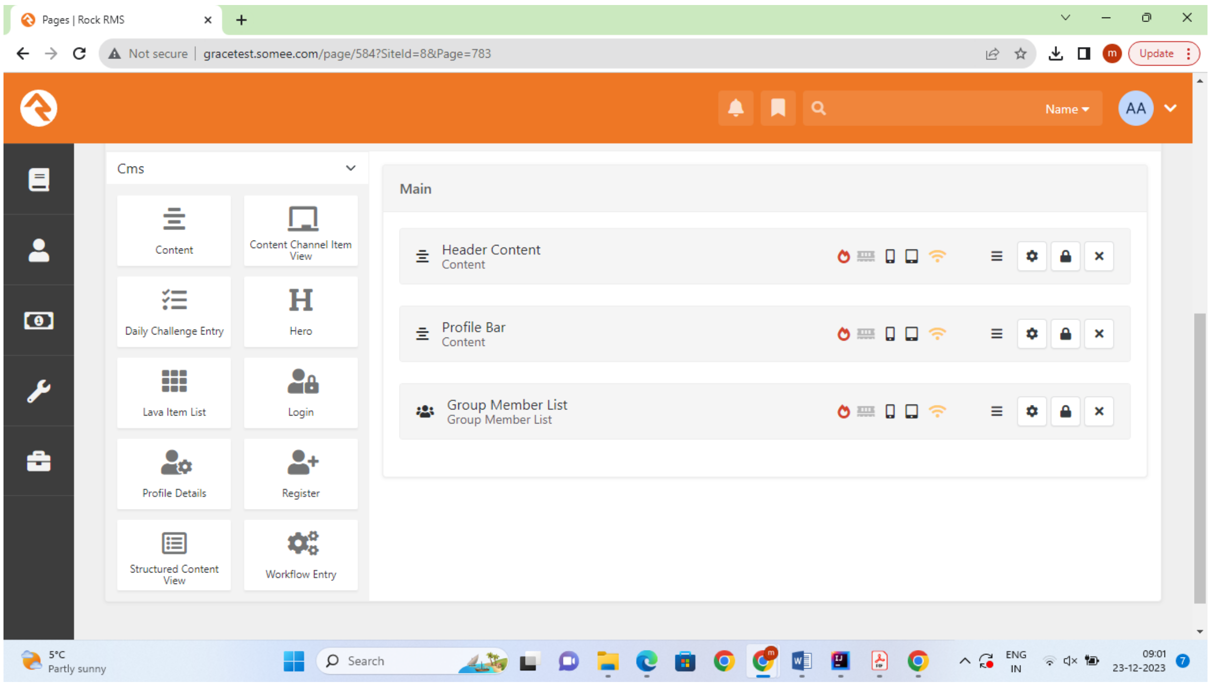
```
{% if Groups == empty %}
    <Rock:NotificationBox NotificationType="Warning" Text="No groups match your search criteria."
/>
{% else %}
    <StackLayout Margin="20">
    <Rock:Divider />
    {% for group in Groups %}
    {% assign distance = Distances[group.Id] %}
    <Grid ColumnDefinitions="1*, 15" ColumnSpacing="12" StyleClass="group-content">
        {% if DetailPage != null %}
            <Grid.GestureRecognizers>
                <TapGestureRecognizer Command="{Binding PushPage}" CommandParameter="{{
DetailPage }}?GroupGuid={{ group.Guid }}" />
            </Grid.GestureRecognizers>
        {% endif %}
        <StackLayout Grid.Column="0" StyleClass="group-primary-content">
            {% if group.Schedule.WeeklyDayOfWeek != null %}
                <Label Text="{{ group.Schedule.WeeklyDayOfWeek }}" StyleClass="group-meeting-day"
/>
            {% endif %}
            <Label Text="{{ group.Name | Escape }}" StyleClass="group-name" />
            <StackLayout Orientation="Horizontal">
                {% if group.Schedule.WeeklyTimeOfDay != null %}
                    <Label Text="Weekly at {{ group.Schedule.WeeklyTimeOfDayText }}"
HorizontalOptions="Start" StyleClass="group-meeting-time" />
                {% elsif group.Schedule != null %}
                    <Label Text="{{ group.Schedule.FriendlyScheduleText }}" HorizontalOptions="Start"
StyleClass="group-meeting-time" />
                {% endif %}
                {% assign topic = group | Attribute:'Topic' %}
                {% if topic != empty %}
                    <Label Text="{{ topic | Escape }}" HorizontalTextAlignment="End"
HorizontalOptions="EndAndExpand" StyleClass="group-topic" />
                {% endif %}
            </StackLayout>
            {% if distance != null %}
                <Label Text="{{ distance | Format:'#,##0.0' }} mi" StyleClass="group-distance" />
            {% endif %}
```

```
          </StackLayout>


          <Rock:Icon IconClass="chevron-right" Grid.Column="1" HorizontalOptions="End"
VerticalOptions="Center" StyleClass="group-more-icon" />
        </Grid>


        <Rock:Divider />
        {% endfor %}
    </StackLayout>
{% endif %}
```

1. **{% if Groups == empty %} ... {% else %} ... {% endif %}**: This is a conditional statement checking if the variable **Groups** is empty. If it is empty, a **Rock:NotificationBox** is displayed with a warning message; otherwise, a **StackLayout** containing a list of groups is rendered.
2. **<StackLayout Margin="20">**: This is a layout container with a margin of 20 units. It contains a list of groups and related information.
3. **{% for group in Groups %} ... {% endfor %}**: This is a loop that iterates over each group in the **Groups** collection.
4. **{% assign distance = Distances[group.Id] %}**: This assigns the distance value associated with the current group's ID from the **Distances** collection to the variable **distance**.
5. **<Grid ColumnDefinitions="1*, 15" ColumnSpacing="12" StyleClass="group-content">**: This creates a grid with two columns, the first taking up 1 part and the second taking up 15 parts. It also adds a style class "group-content" and sets column spacing.
6. **{% if DetailPage != null %} ... {% endif %}**: This is a conditional statement checking if **DetailPage** is not null. If true, a tap gesture recognizer is added to the grid for navigation purposes.
7. **<StackLayout Grid.Column="0" StyleClass="group-primary-content">**: This is a nested stack layout within the grid's first column, containing information about the group.
8. Various **<Label>** elements are used to display information about the group, such as its name, meeting day, meeting time, topic, and distance.
9. **<Rock:Icon IconClass="chevron-right" Grid.Column="1" HorizontalOptions="End" VerticalOptions="Center" StyleClass="group-more-icon" />**: This adds an icon to the second column of the grid, indicating that there is more information or an action available.
10. **<Rock:Divider />**: This is a custom divider element, possibly used for visual separation between groups.

The code overall renders a list of groups with associated information, and if there are no groups, it displays a warning message. The templating language used here appears to include conditional statements, loops, and variable assignments for dynamic content generation.

Group Members:



The below code and explanation are of block Header Content

```
<StackLayout>
  <Grid>

      <Image Source="https://assets-global.website-
files.com/5f6b9a421d5a61e1d0cd9e3d/62f285c4f9aa3441840257d6_nathan-mullet-
pmiW630yDPE-unsplash.jpg"
          Aspect="AspectFill"  />

      <BoxView BackgroundColor="#80000000"/>

      <Label Text="Groups"
          TextColor="White"
          FontSize="36"
          FontAttributes="Bold"
          HorizontalOptions="Center"
          VerticalOptions="Center" />
  </Grid>
</StackLayout>
```

Explanation:

1. **<StackLayout>**: This is a layout container that stacks its child elements vertically by default. It's a common layout container in Xamarin.Forms.
2. **<Grid>**: This is a layout container that allows you to create a grid-based layout. It is used to arrange child elements in rows and columns.
3. **<Image>**: An **Image** element is used to display an image. In this case, it is displaying an image from the specified URL. The **Aspect="AspectFill"** property ensures that the image fills the available space while maintaining its aspect ratio.
4. **<BoxView BackgroundColor="#80000000"/>**: This is a **BoxView** element, which is a simple rectangular box that can be used for various purposes. In this case, it serves as an overlay with a semi-transparent black background (**#80000000** represents a dark translucent background color).
5. **<Label>**: This is a **Label** element displaying the text "Groups." It is styled with white text color, a font size of 36, and bold font attributes. The **HorizontalOptions** and **VerticalOptions** properties are set to "Center," ensuring that the label is centered within the grid.

Overall, the code creates a visually appealing layout with an image background, a semi-transparent overlay, and a centered label displaying the text "Groups." It's a common approach for creating introductory or splash screens in mobile applications. The **<Grid>** allows for a more complex layout by overlaying the image, box view, and label in a structured way.

The below code and explanation are of block Profile bar:

```
<StackLayout Margin="0,0,0,0" Orientation="Vertical" VerticalOptions="Center"
BackgroundColor="#7bb446">

  <StackLayout Margin="0,10,0,10" Orientation="Horizontal" HorizontalOptions="Start">

    <Rock:Icon IconClass="user" StyleClass="text-primary" Margin="10,0,0,0" TextColor="White"
VerticalOptions="CenterAndExpand" FontSize="24" />

    <Label Grid.Column="1" StyleClass="pl-3" Text="Hi {{ CurrentPerson.FirstName }}!"
TextColor="White" Margin="0,10,0,10" VerticalOptions="CenterAndExpand">

      <Label.GestureRecognizers>

        <TapGestureRecognizer Command="{Binding PushPage}" CommandParameter="09ee7ee8-
0b44-4b88-b5ca-4dc26dec006f" />

      </Label.GestureRecognizers>

    </Label>

  </StackLayout>



  <Rock:Divider />

</StackLayout>
```

Explanation:

1. **<StackLayout Margin="0,0,0,0" Orientation="Vertical" VerticalOptions="Center" BackgroundColor="#7bb446">**: This is the outermost **StackLayout** that serves as a container for the entire layout. It has a green background color (#7bb446) and is vertically centered on the screen.
2. **<StackLayout Margin="0,10,0,10" Orientation="Horizontal" HorizontalOptions="Start">**: This is a nested **StackLayout** within the outer one. It is oriented horizontally and has a margin of 10 units at the top and bottom. The **HorizontalOptions** is set to "Start," aligning its content to the left.
3. **<Rock:Icon IconClass="user" StyleClass="text-primary" Margin="10,0,0,0" TextColor="White" VerticalOptions="CenterAndExpand" FontSize="24" />**: This is a custom **Rock:Icon** control displaying a user icon. It has a white color, a margin to the left, and a font size of 24.
4. **<Label Grid.Column="1" StyleClass="pl-3" Text="Hi {{ CurrentPerson.FirstName }}!" TextColor="White" Margin="0,10,0,10" VerticalOptions="CenterAndExpand">**: This is a **Label** displaying a greeting message. It uses data binding (**{{ CurrentPerson.FirstName }}**) to dynamically display the user's first name. It has a white text color, a margin, and is vertically centered and expanded.
5. **<Label.GestureRecognizers>**: This block contains a **TapGestureRecognizer** that is associated with the **Label**. It binds to the "PushPage" command in the view model, passing a specific parameter (ID) when tapped.
6. **<Rock:Divider />**: This is a custom **Rock:Divider** control, which is likely a visual separator/divider element.

Overall, the code represents a user interface with a green background, a user icon, a greeting label with a dynamic first name, and a divider below it. The **TapGestureRecognizer** allows an action to be triggered when the label is tapped, and the custom divider adds visual separation to the layout.

The below code and explanation are of block Group Member list:

```
<StackLayout StyleClass="members-container"
  Spacing="0">
  {% for member in Members %}
    <Frame StyleClass="member-container"
      Margin="0"
      BackgroundColor="White"
      HasShadow="false"
      HeightRequest="40">
        <StackLayout Orientation="Horizontal"
          Spacing="0"
```

```
VerticalOptions="Center">
<Rock:Image Source="{{ member.PhotoUrl | Escape }}"
  StyleClass="member-person-image"
  VerticalOptions="Start"
  Aspect="AspectFit"
  Margin="0, 4, 14, 0"
  BackgroundColor="#e4e4e4">
  <Rock:CircleTransformation />
</Rock:Image>

<StackLayout Spacing="0"
  HorizontalOptions="FillAndExpand">
  <StackLayout Orientation="Horizontal"
  VerticalOptions="Center">
    <Label StyleClass="member-name"
      Text="{{ member.FullName }}"
      LineBreakMode="TailTruncation"
      HorizontalOptions="FillAndExpand" />

    <Grid ColumnSpacing="4"
      RowSpacing="0"
      ColumnDefinitions="*, Auto"
      VerticalOptions="Start">

      <Rock:Icon IconClass="chevron-right"
        VerticalTextAlignment="Start"
        Grid.Column="1"
        StyleClass="note-read-more-icon"
        />
    </Grid>
  </StackLayout>
```

```
                    <Label StyleClass="member-text"
                       Grid.Column="0"
                       MaxLines="2"
                       LineBreakMode="TailTruncation"
                       Text="{{ member.GroupRole | Escape }}" />
                </StackLayout>
            </StackLayout>
            <Frame.GestureRecognizers>
                <TapGestureRecognizer Command="{Binding PushPage}"
                 CommandParameter="{{ DetailPage }}?EntitySetGuid={{ member.Guid }}" />
            </Frame.GestureRecognizers>
        </Frame>
    <BoxView HorizontalOptions="FillAndExpand"
        HeightRequest="1"
        Color="#cccccc" />
  {% endfor %}
</StackLayout>
```

Explanation:

1. **<StackLayout StyleClass="members-container" Spacing="0">**: The outermost **StackLayout** is used as a container for the list of members. It has a style class and zero spacing between its children.
2. **{% for member in Members %} ... {% endfor %}**: This is a loop that iterates over each member in the **Members** collection.
3. **<Frame StyleClass="member-container" Margin="0" BackgroundColor="White" HasShadow="false" HeightRequest="40">**: Each member is enclosed in a **Frame** to provide a visual container with specific styling. It has a white background color, no shadow, and a fixed height.
4. **<StackLayout Orientation="Horizontal" Spacing="0" VerticalOptions="Center">**: A horizontal **StackLayout** is used to arrange the member information side by side.
5. **<Rock:Image Source="{{ member.PhotoUrl | Escape }}" StyleClass="member-person-image" ...>**: A custom **Rock:Image** is used to display the member's photo. It applies a circular transformation.
6. **<StackLayout Spacing="0" HorizontalOptions="FillAndExpand">**: Another nested **StackLayout** contains details about the member.

7. **<StackLayout Orientation="Horizontal" VerticalOptions="Center"> ... </StackLayout>**: This horizontal **StackLayout** includes the member's full name and an icon.

8. **<Grid ColumnSpacing="4" RowSpacing="0" ColumnDefinitions="*, Auto" VerticalOptions="Start">**: A **Grid** is used to arrange an icon (chevron-right) in the horizontal layout.

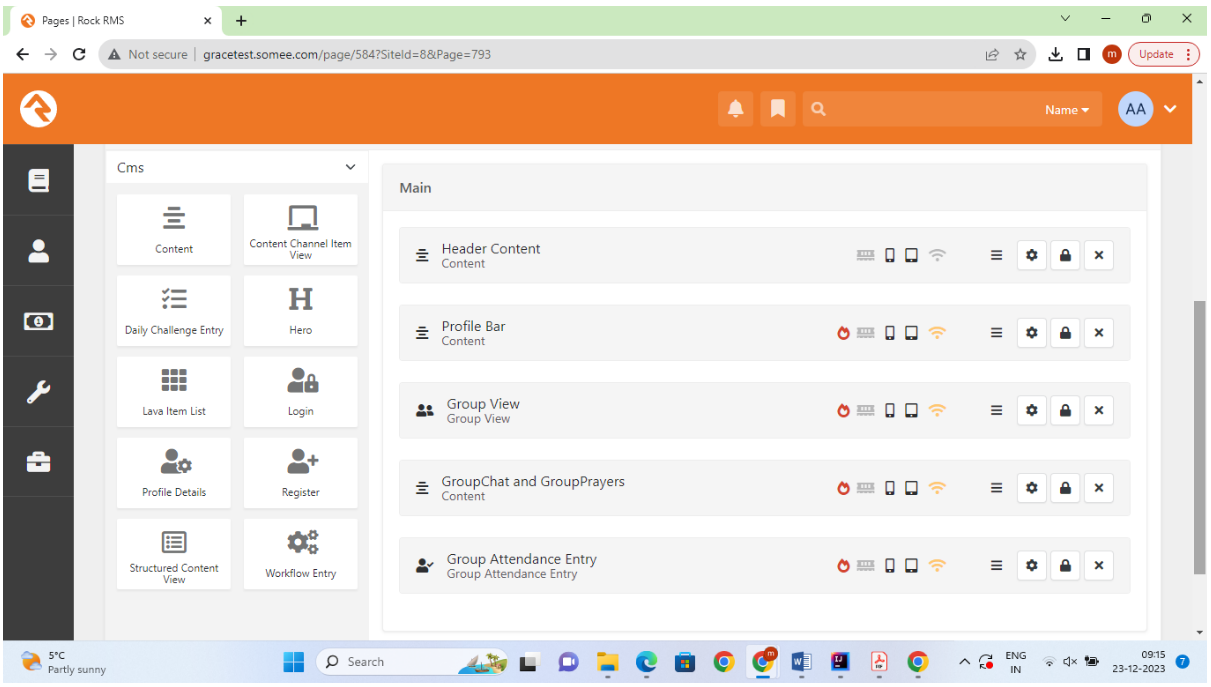
9. **<Label StyleClass="member-text" Grid.Column="0" MaxLines="2" LineBreakMode="TailTruncation" Text="{{ member.GroupRole | Escape }}" />**: A **Label** is used to display the member's role within a group.

10. **<Frame.GestureRecognizers>**: A **TapGestureRecognizer** is added to the **Frame** to handle the tap gesture, triggering a command in the view model with a specific parameter.

11. **<BoxView HorizontalOptions="FillAndExpand" HeightRequest="1" Color="#cccccc" />**: A **BoxView** is added to create a horizontal line as a separator between member entries.

The overall code is a template for rendering a list of members with their photos, names, roles, and a navigation action on tap. Each member entry is visually contained within a **Frame**, and a horizontal line separates each member entry.

Group:



Main

Header Content
Content

Profile Bar
Content

Group View
Group View

GroupChat and GroupPrayers
Content

Group Attendance Entry
Group Attendance Entry

The below code and explanation are of block Header Content:

<StackLayout>

<Grid>

<Image Source="https://assets-global.website-files.com/5f6b9a421d5a61e1d0cd9e3d/62f285c4f9aa3441840257d6_nathan-mullet-pmiW630yDPE-unsplash.jpg"

```
        Aspect="AspectFill"  />


    <BoxView BackgroundColor="#80000000"/>


    <Label Text="Groups"
        TextColor="White"
        FontSize="36"
        FontAttributes="Bold"
        HorizontalOptions="Center"
        VerticalOptions="Center" />
  </Grid>
</StackLayout>
```

Explanation:

1. **<StackLayout>**: This is a layout container that stacks its child elements vertically by default. It's a common layout container in Xamarin.Forms.
2. **<Grid>**: This is a layout container that allows you to create a grid-based layout. It is used to arrange child elements in rows and columns.
3. **<Image>**: An **Image** element is used to display an image. In this case, it is displaying an image from the specified URL. The **Aspect="AspectFill"** property ensures that the image fills the available space while maintaining its aspect ratio.
4. **<BoxView BackgroundColor="#80000000"/>**: This is a **BoxView** element, which is a simple rectangular box that can be used for various purposes. In this case, it serves as an overlay with a semi-transparent black background (**#80000000** represents a dark translucent background color).
5. **<Label>**: This is a **Label** element displaying the text "Groups." It is styled with white text color, a font size of 36, and bold font attributes. The **HorizontalOptions** and **VerticalOptions** properties are set to "Center," ensuring that the label is centered within the grid.

Overall, the code creates a visually appealing layout with an image background, a semi-transparent overlay, and a centered label displaying the text "Groups." This type of layout is commonly used for introductory or splash screens in mobile applications. The **<Grid>** allows for a more complex layout by overlaying the image, box view, and label in a structured way.

The below code and explanation are of block Profile bar:

```
<StackLayout Margin="0,0,0,0" Orientation="Vertical" VerticalOptions="Center"
BackgroundColor="#7bb446">

  <StackLayout Margin="0,10,0,10" Orientation="Horizontal" HorizontalOptions="Start">
```

```
        <Rock:Icon IconClass="user" StyleClass="text-primary" Margin="10,0,0,0" TextColor="White"
VerticalOptions="CenterAndExpand" FontSize="24" />

        <Label Grid.Column="1" StyleClass="pl-3" Text="Hi {{ CurrentPerson.FirstName }}!"
TextColor="White" Margin="0,10,0,10" VerticalOptions="CenterAndExpand">

            <Label.GestureRecognizers>

                <TapGestureRecognizer Command="{Binding PushPage}" CommandParameter="09ee7ee8-
0b44-4b88-b5ca-4dc26dec006f" />

            </Label.GestureRecognizers>

        </Label>

    </StackLayout>


    <Rock:Divider />

</StackLayout>
```

Explanation:

1. **<StackLayout Margin="0,0,0,0" Orientation="Vertical" VerticalOptions="Center" BackgroundColor="#7bb446">**: This is the outermost **StackLayout** that serves as a container for the entire layout. It has a green background color (#7bb446) and is vertically centered on the screen.
2. **<StackLayout Margin="0,10,0,10" Orientation="Horizontal" HorizontalOptions="Start">**: This is a nested **StackLayout** within the outer one. It is oriented horizontally and has a margin of 10 units at the top and bottom. The **HorizontalOptions** is set to "Start," aligning its content to the left.
3. **<Rock:Icon IconClass="user" StyleClass="text-primary" Margin="10,0,0,0" TextColor="White" VerticalOptions="CenterAndExpand" FontSize="24" />**: This is a custom **Rock:Icon** control displaying a user icon. It has a white color, a margin to the left, and a font size of 24.
4. **<Label Grid.Column="1" StyleClass="pl-3" Text="Hi {{ CurrentPerson.FirstName }}!" TextColor="White" Margin="0,10,0,10" VerticalOptions="CenterAndExpand">**: This is a **Label** displaying a greeting message. It uses data binding (**{{ CurrentPerson.FirstName }}**) to dynamically display the user's first name. It has a white text color, a margin, and is vertically centered and expanded.
5. **<Label.GestureRecognizers>**: This block contains a **TapGestureRecognizer** that is associated with the **Label**. It binds to the "PushPage" command in the view model, passing a specific parameter (ID) when tapped.
6. **<Rock:Divider />**: This is a custom **Rock:Divider** control, which is likely a visual separator/divider element.

Overall, the code represents a user interface with a green background, a user icon, a greeting label with a dynamic first name, and a divider below it. The **TapGestureRecognizer** allows an action to be triggered when the label is tapped, and the custom divider adds visual separation to the layout.

The below code and explanation are of block Group View:

```
<StackLayout Spacing="0" Margin="20">
  {% assign groupMemberCount = Group.Members | Size %}


  <StackLayout Orientation="Horizontal" Spacing="20">
    <StackLayout Orientation="Vertical" Spacing="0" HorizontalOptions="FillAndExpand">
      <Label StyleClass="h1" Text="{{ Group.Name | Escape }} Group" />
      <Label StyleClass="text" Text="{{ 'member' | ToQuantity:groupMemberCount }}" />
    </StackLayout>


    {% if GroupEditPage != '' and AllowedActions.Edit == true %}
      <Rock:Icon IconClass="Ellipsis-v" FontSize="24" TextColor="#ccc" Command="{Binding
ShowActionPanel}" Margin="0,8,4,0">
        <Rock:Icon.CommandParameter>
          <Rock:ShowActionPanelParameters Title="Group Actions" CancelTitle="Cancel">
            <Rock:ActionPanelButton Title="Edit Group"
              Command="{Binding PushPage}" CommandParameter="{{ GroupEditPage
}}?GroupGuid={{ Group.Guid }}" />
          </Rock:ShowActionPanelParameters>
        </Rock:Icon.CommandParameter>
      </Rock:Icon>
    {% endif %}
  </StackLayout>


  <Rock:Divider StyleClass="my-24" />


  <!-- Handle Group Attributes -->
  {% if VisibleAttributes != empty %}
```

```
        <Rock:ResponsiveLayout ColumnSpacing="0">
            {% for attribute in VisibleAttributes %}
                {% if attribute.FormattedValue != '' %}
                    <Rock:ResponsiveColumn ExtraSmall="6">
                        <Rock:FieldContainer>
                            <Rock:Literal Label="{{ attribute.Name | Escape }}" Text="{{
attribute.FormattedValue }}" />
                        </Rock:FieldContainer>
                    </Rock:ResponsiveColumn>
                {% endif %}
            {% endfor %}
        </Rock:ResponsiveLayout>
    {% endif %}

    <!-- Handle displaying of leaders -->
    {% if ShowLeaderList == true %}
        <Label StyleClass="title, mt-24" Text="Leaders" />
        <Rock:ResponsiveLayout ColumnSpacing="0">
            {% assign members = Group.Members | OrderBy:'Person.FullName' %}
            {% for member in members %}
                {% if member.GroupRole.IsLeader == false %}{% continue %}{% endif %}

                    <Rock:ResponsiveColumn ExtraSmall="6">
                        <Label Text="{{ member.Person.FullName }}" />
                    </Rock:ResponsiveColumn>
                    <Rock:ResponsiveColumn ExtraSmall="6">
                        <Label StyleClass="o-60" Text="{{ member.GroupRole.Name }}" />
                    </Rock:ResponsiveColumn>

            {% endfor %}
        </Rock:ResponsiveLayout>
```

```
  {% endif %}


  <Rock:Divider StyleClass="my-24" />
</StackLayout>
```

Explanation:

1. **<StackLayout Spacing="0" Margin="20">**: The outermost **StackLayout** serves as a container for the entire layout. It has zero spacing between its children and a margin of 20 units.
2. **{% assign groupMemberCount = Group.Members | Size %}**: The total number of members in the group is assigned to the variable **groupMemberCount**.
3. **<!-- Header Section --> ... <!-- Edit Icon (if allowed) -->**: This section displays information about the group, including its name and member count. It also includes an edit icon if the user has permission to edit the group.
4. **<Rock:Divider StyleClass="my-24" />**: A custom **Rock:Divider** is used to create a visual separation with a margin of 24 units.
5. **{% if VisibleAttributes != empty %} ... {% endif %}**: This section handles the display of group attributes. It uses a **ResponsiveLayout** to dynamically arrange the attributes in columns.
6. **{% if ShowLeaderList == true %} ... {% endif %}**: This section handles the display of leaders in the group. It uses a **ResponsiveLayout** to show leader names and roles in separate columns.
7. **<Rock:Divider StyleClass="my-24" />**: Another custom **Rock:Divider** is added for visual separation.

Overall, the code is a template for rendering detailed information about a group, including its name, member count, attributes, and leaders. The templating language includes conditional checks and looping constructs for dynamic rendering based on data availability and user permissions.

The below code and explanation are of block Group chat and Group prayers:

```
<StackLayout>
  <Grid>
    <Grid.RowDefinitions>
      <RowDefinition Height="Auto"/>
      <RowDefinition Height="Auto"/>
    </Grid.RowDefinitions>


    <Grid.ColumnDefinitions>
      <ColumnDefinition Width="*"/>
      <ColumnDefinition Width="*"/>
    </Grid.ColumnDefinitions>



    <Button
      x:Name="GrouchatButton"
      Text="Group Chat"
      StyleClass="btn, btn-primary"
      HorizontalOptions="Center"
      BackgroundColor="#000000"
      Command="{Binding PushPage}"
      CommandParameter="dfbf89b6-4ed8-4770-8b02-905d7b08d248"
      CornerRadius="8"
      WidthRequest="200"
      HeightRequest="40"
      FontAttributes="Bold"
      FontFamily="Segoe UI"
      Grid.Row="0"
      Grid.Column="0"
      Margin="20,25,10,0" />

    <Button
      x:Name="GroupPrayersButton"
      Text="Group Prayers"
      StyleClass="btn, btn-primary"
      HorizontalOptions="Center"
      BackgroundColor="#000000"
      Command="{Binding PushPage}"
      CommandParameter="634c5764-5f7b-4169-a084-6793c2a930fd"
      CornerRadius="8"
      WidthRequest="200"
```

```
        HeightRequest="40"
        FontAttributes="Bold"
        FontFamily="Segoe UI"
        Grid.Row="0"
        Grid.Column="1"
        Margin="10,25,20,0" />

         <Button
        x:Name="LeaderToolboxButton"
        Text="Leader Toolbox"
        StyleClass="btn, btn-primary"
        HorizontalOptions="Center"
        BackgroundColor="#7bb446"
        Command="{Binding PushPage}"
        CommandParameter="a9342a39-67f2-4ae2-b5d8-4ddc324c91e5"
        CornerRadius="8"
        WidthRequest="300"  HeightRequest="60"   FontSize="18"     FontAttributes="Bold"
        FontFamily="Segoe UI"
        Grid.Row="1"
        Grid.ColumnSpan="2" />

    </Grid>
    <Rock:Divider StyleClass="my-24" />
</StackLayout>
```

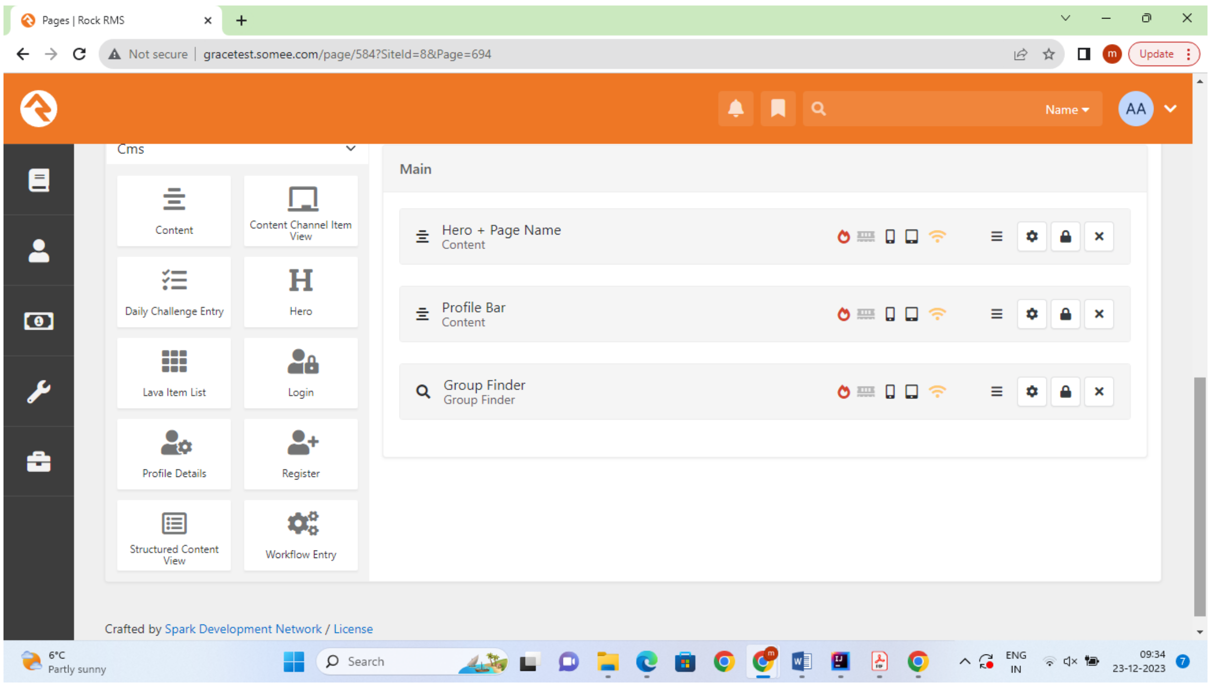
Explanation:

1. **<StackLayout>**: This is the outermost layout container that stacks its child elements vertically.
2. **<Grid>**: This layout container is used to arrange child elements in rows and columns.
3. **<!-- Define Rows and Columns -->**: The **Grid** defines two rows and two columns with proportional widths and heights.

4. **<Button x:Name="GrouchatButton" ...>**: This is the first button named "GrouchatButton." It triggers a command (**PushPage**) when clicked and has various styling properties like background color, corner radius, and margins.
5. **<Button x:Name="GroupPrayersButton" ...>**: This is the second button named "GroupPrayersButton" with similar properties as the first button.
6. **<Button x:Name="LeaderToolboxButton" ...>**: This is the third button named "LeaderToolboxButton." It has a different background color, larger width and height, and spans across both columns in the second row.
7. **<Rock:Divider StyleClass="my-24" />**: This is a custom **Rock:Divider** control, possibly used for visual separation. It has a style class for styling and a margin of 24 units.

Overall, the code represents a layout with three buttons arranged in a grid, each with specific styling and associated commands. The custom divider adds visual spacing at the bottom of the layout.

<u>Teams:</u>



The below code and explanation are of block Hero+Page name:

```
<StackLayout>

   <Grid>


      <Image Source="https://assets-global.website-
files.com/5f6b9a421d5a61e1d0cd9e3d/62f285c4f9aa3441840257d6_nathan-mullet-pmiW630yDPE-
unsplash.jpg"

            Aspect="AspectFill"  />
```

```
    <BoxView BackgroundColor="#80000000"/>


    <Label Text="Teams"

        TextColor="White"

        FontSize="36"

        FontAttributes="Bold"

        HorizontalOptions="Center"

        VerticalOptions="Center" />

  </Grid>

</StackLayout>
```

Explanation:

1. **<StackLayout>**: This is the outermost layout container that stacks its child elements vertically.
2. **<Grid>**: This layout container is used to arrange child elements in rows and columns.
3. **<Image Source="...">**: This is an **Image** element displaying an image from the specified URL. The **Aspect="AspectFill"** property ensures that the image fills the available space while maintaining its aspect ratio.
4. **<BoxView BackgroundColor="#80000000"/>**: This is a **BoxView** element, which is a simple rectangular box that can be used for various purposes. In this case, it serves as an overlay with a semi-transparent black background (**#80000000** represents a dark translucent background color).
5. **<Label>**: This is a **Label** element displaying the text "Teams." It is styled with white text color, a font size of 36, and bold font attributes. The **HorizontalOptions** and **VerticalOptions** properties are set to "Center," ensuring that the label is centered within the grid.

Overall, the code creates a visually appealing layout with an image background, a semi-transparent overlay, and a centered label displaying the text "Teams." This type of layout is commonly used for introductory or splash screens in mobile applications. The **<Grid>** allows for a more complex layout by overlaying the image, box view, and label in a structured way.

The below code and explanation are of block Profile bar:

```
<StackLayout Margin="0,0,0,0" Orientation="Vertical" VerticalOptions="Center"
BackgroundColor="#7bb446">

  <StackLayout Margin="0,10,0,10" Orientation="Horizontal" HorizontalOptions="Start">

    <Rock:Icon IconClass="user" StyleClass="text-primary" Margin="10,0,0,0" TextColor="White"
VerticalOptions="CenterAndExpand" FontSize="24" />

    <Label Grid.Column="1" StyleClass="pl-3" Text="Hi {{ CurrentPerson.FirstName }}!"
TextColor="White" Margin="0,10,0,10" VerticalOptions="CenterAndExpand">
```

```
        <Label.GestureRecognizers>

            <TapGestureRecognizer Command="{Binding PushPage}" CommandParameter="09ee7ee8-
0b44-4b88-b5ca-4dc26dec006f" />

        </Label.GestureRecognizers>

      </Label>

  </StackLayout>


  <Rock:Divider />

</StackLayout>
```

Explanation:

1. **<StackLayout>**: This is the outermost layout container that stacks its child elements vertically.
2. **<Grid>**: This layout container is used to arrange child elements in rows and columns.
3. **<Image Source="...">**: This is an **Image** element displaying an image from the specified URL. The **Aspect="AspectFill"** property ensures that the image fills the available space while maintaining its aspect ratio.
4. **<BoxView BackgroundColor="#80000000"/>**: This is a **BoxView** element, which is a simple rectangular box that can be used for various purposes. In this case, it serves as an overlay with a semi-transparent black background (**#80000000** represents a dark translucent background color).
5. **<Label>**: This is a **Label** element displaying the text "Teams." It is styled with white text color, a font size of 36, and bold font attributes. The **HorizontalOptions** and **VerticalOptions** properties are set to "Center," ensuring that the label is centered within the grid.

Overall, the code creates a visually appealing layout with an image background, a semi-transparent overlay, and a centered label displaying the text "Teams." This type of layout is commonly used for introductory or splash screens in mobile applications. The **<Grid>** allows for a more complex layout by overlaying the image, box view, and label in a structured way.

The below code and explanation are of block Group Finder:

```
{% if Groups == empty %}

  <Rock:NotificationBox NotificationType="Warning" Text="No groups match your search criteria."
/>

{% else %}

  <StackLayout Margin="20">

    <Rock:Divider />

    {% for group in Groups %}

    {% assign distance = Distances[group.Id] %}
    <Grid ColumnDefinitions="1*, 15" ColumnSpacing="12" StyleClass="group-content">
      {% if DetailPage != null %}
        <Grid.GestureRecognizers>
          <TapGestureRecognizer Command="{Binding PushPage}" CommandParameter="{{
DetailPage }}?GroupGuid={{ group.Guid }}" />
        </Grid.GestureRecognizers>
      {% endif %}
      <StackLayout Grid.Column="0" StyleClass="group-primary-content">
        {% if group.Schedule.WeeklyDayOfWeek != null %}
          <Label Text="{{ group.Schedule.WeeklyDayOfWeek }}" StyleClass="group-meeting-day"
/>
        {% endif %}
        <Label Text="{{ group.Name | Escape }}" StyleClass="group-name" />
        <StackLayout Orientation="Horizontal">
          {% if group.Schedule.WeeklyTimeOfDay != null %}
            <Label Text="Weekly at {{ group.Schedule.WeeklyTimeOfDayText }}"
HorizontalOptions="Start" StyleClass="group-meeting-time" />
          {% elsif group.Schedule != null %}
            <Label Text="{{ group.Schedule.FriendlyScheduleText }}" HorizontalOptions="Start"
StyleClass="group-meeting-time" />
          {% endif %}
          {% assign topic = group | Attribute:'Topic' %}
          {% if topic != empty %}
            <Label Text="{{ topic | Escape }}" HorizontalTextAlignment="End"
HorizontalOptions="EndAndExpand" StyleClass="group-topic" />
          {% endif %}
        </StackLayout>
        {% if distance != null %}
          <Label Text="{{ distance | Format:'#,##0.0' }} mi" StyleClass="group-distance" />
        {% endif %}
      </StackLayout>
```

```
            <Rock:Icon IconClass="chevron-right" Grid.Column="1" HorizontalOptions="End"
VerticalOptions="Center" StyleClass="group-more-icon" />

        </Grid>


        <Rock:Divider />

        {% endfor %}

    </StackLayout>

{% endif %}
```

Explanation:

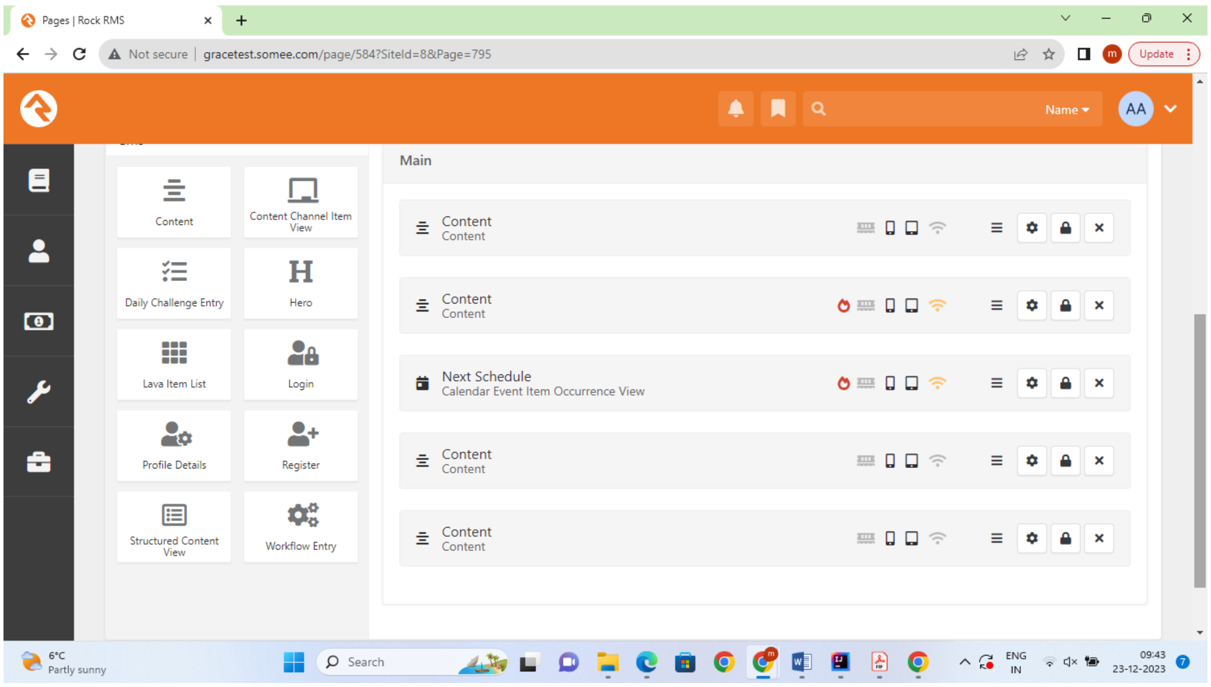
1. **{% if Groups == empty %}**: This is a conditional statement checking if the variable **Groups** is empty. If true, it displays a warning notification using a custom **Rock:NotificationBox** with the text "No groups match your search criteria."
2. **{% else %}**: If the **Groups** variable is not empty, it proceeds to the else block.
3. **<StackLayout Margin="20">**: This is a layout container (probably representing a vertical stack) with a margin of 20 units.
4. **<Rock:Divider />**: This is a custom divider, possibly used for visual separation.
5. **{% for group in Groups %}**: This is a loop that iterates over each item in the **Groups** collection.
6. **{% assign distance = Distances[group.Id] %}**: Assigns the distance for the current group from the **Distances** collection.
7. **<Grid ColumnDefinitions="1*, 15" ColumnSpacing="12" StyleClass="group-content">**: This sets up a grid with two columns, the first taking up 1 part and the second taking up 15 parts. It also adds a style class "group-content."
8. **{% if DetailPage != null %}**: Checks if **DetailPage** is not null.
9. **<Grid.GestureRecognizers>**: Adds a tap gesture recognizer to the grid if **DetailPage** is not null. The tap gesture triggers the **PushPage** command with the **GroupGuid** as a parameter.
10. **<StackLayout Grid.Column="0" StyleClass="group-primary-content">**: This is a nested stack layout that contains information about the group. It has a style class "group-primary-content."
11. Various **<Label>** elements: Display information about the group, such as meeting day, name, meeting time, topic, and distance.
12. **<Rock:Icon IconClass="chevron-right" Grid.Column="1" HorizontalOptions="End" VerticalOptions="Center" StyleClass="group-more-icon" />**: This is an icon (possibly a chevron) positioned at the end of the grid (second column) with style class "group-more-icon."
13. **<Rock:Divider />**: Another custom divider, possibly used for visual separation between groups.

14. **{% endfor %}**: Ends the loop over the **Groups** collection.

15. **{% endif %}**: Ends the conditional block.

Overall, this code generates a layout that displays information about groups, including meeting details, names, topics, distances, and an icon to indicate additional details. If no groups match the search criteria, a warning notification is displayed. The layout is styled using custom classes and dividers for visual appeal.

Team:



The below code and explanation are of block Content:

```
<StackLayout>
  <Grid>
    <Grid.RowDefinitions>
      <RowDefinition Height="Auto" />
    </Grid.RowDefinitions>


    <Image
      x:Name="ml"
      Source="https://www.50sundays.com/wp-content/uploads/2015/01/50-sundays-promo-
shot-at-symmes.jpg"
      HorizontalOptions="CenterAndExpand"
      VerticalOptions="CenterAndExpand" />
```

```
        <Label
            Text="Team"
            HorizontalOptions="CenterAndExpand"
            VerticalOptions="Center"
            FontFamily="Poppins"
            FontAttributes="Bold"
            BackgroundColor="Transparent"
            TextColor="White"
            FontSize="26" />
    </Grid>
</StackLayout>
```

Explanation:

1. **<StackLayout>**: This is the outermost layout container that stacks its child elements vertically.
2. **<Grid>**: This layout container is used to arrange child elements in rows and columns.
3. **<Grid.RowDefinitions>**: Defines the row structure for the grid. In this case, it has a single row with a height set to "Auto," meaning it will adjust its height based on the content.
4. **<Image x:Name="ml" Source="...">**: This is an **Image** element displaying an image from the specified URL. It has the name "ml" assigned to it (**x:Name="ml"**). The **HorizontalOptions** and **VerticalOptions** properties are set to "CenterAndExpand," ensuring that the image is centered both horizontally and vertically within the grid.
5. **<Label Text="Team" ...>**: This is a **Label** element displaying the text "Team." It has various properties set, such as **HorizontalOptions** and **VerticalOptions** for centering, **FontFamily** set to "Poppins," **FontAttributes** set to "Bold," **BackgroundColor** set to "Transparent," **TextColor** set to white, and **FontSize** set to 26.

Overall, the code creates a simple layout with an image and a label centered within a grid. The image is loaded from a URL, and the label displays the text "Team" with specified styling properties. This type of layout is commonly used for displaying introductory or branding content in mobile applications.

The below code and explanation are of block Content:

```
<StackLayout Margin="0,0,0,0" Orientation="Vertical" VerticalOptions="Center"
BackgroundColor="#7bb446">

    <StackLayout Margin="0,10,0,10" Orientation="Horizontal" HorizontalOptions="Start">

        <Rock:Icon IconClass="user" StyleClass="text-primary" Margin="10,0,0,0" TextColor="White"
VerticalOptions="CenterAndExpand" FontSize="24" />
```

```
        <Label Grid.Column="1" StyleClass="pl-3" Text="Hi {{ CurrentPerson.FirstName }}!"
TextColor="White" Margin="0,10,0,10" VerticalOptions="CenterAndExpand">

          <Label.GestureRecognizers>

            <TapGestureRecognizer Command="{Binding PushPage}" CommandParameter="09ee7ee8-
0b44-4b88-b5ca-4dc26dec006f" />

          </Label.GestureRecognizers>

        </Label>

    </StackLayout>


    <Rock:Divider />

</StackLayout>
```

Explanation:

1. **<StackLayout Margin="0,0,0,0" Orientation="Vertical" VerticalOptions="Center" BackgroundColor="#7bb446">**: This is the outermost **StackLayout** that serves as a container for the entire layout. It has a green background color (#7bb446) and is vertically centered on the screen.
2. **<StackLayout Margin="0,10,0,10" Orientation="Horizontal" HorizontalOptions="Start">**: This is a nested **StackLayout** within the outer one. It is oriented horizontally, has a margin of 10 units at the top and bottom, and aligns its content to the left.
3. **<Rock:Icon IconClass="user" StyleClass="text-primary" Margin="10,0,0,0" TextColor="White" VerticalOptions="CenterAndExpand" FontSize="24" />**: This is a custom **Rock:Icon** control displaying a user icon. It has a white color, a margin to the left, and a font size of 24.
4. **<Label Grid.Column="1" StyleClass="pl-3" Text="Hi {{ CurrentPerson.FirstName }}!" TextColor="White" Margin="0,10,0,10" VerticalOptions="CenterAndExpand">**: This is a **Label** displaying a greeting message. It uses data binding (**{{ CurrentPerson.FirstName }}**) to dynamically display the user's first name. It has a white text color, a margin, and is vertically centered and expanded.
5. **<Label.GestureRecognizers>**: This block contains a **TapGestureRecognizer** that is associated with the **Label**. It binds to the "PushPage" command in the view model, passing a specific parameter (ID) when tapped.
6. **<Rock:Divider />**: This is a custom **Rock:Divider** control, which is likely a visual separator/divider element.

Overall, the code represents a user interface with a green background, a user icon, a greeting label with a dynamic first name, and a divider below it. The **TapGestureRecognizer** allows an action to be triggered when the label is tapped, and the custom divider adds visual separation to the layout.

The below code and explanation are of block Content:

```
<Grid>
  <Grid.RowDefinitions>
    <RowDefinition Height="Auto"/>
    <RowDefinition Height="Auto"/>
  </Grid.RowDefinitions>

  <Grid.ColumnDefinitions>
    <ColumnDefinition Width="*"/>
    <ColumnDefinition Width="*"/>
  </Grid.ColumnDefinitions>


  <Button
    x:Name="MyScheduleButton"
    Text="My Schedule"
    StyleClass="btn, btn-primary"
    HorizontalOptions="Center"
    BackgroundColor="#000000"
    Command="{Binding PushPage}"
    CommandParameter="79099eb8-ce5b-4b59-b5ec-e3829a65f900"
    CornerRadius="8"
    WidthRequest="200"
    HeightRequest="40"
    FontAttributes="Bold"
    FontFamily="Segoe UI"
    Grid.Row="0"
    Grid.Column="0"
    Margin="20,25,10,0" />

  <Button
```

```
        x:Name="GroupChatButton"
        Text="Group Chat"
        StyleClass="btn, btn-primary"
        HorizontalOptions="Center"
        BackgroundColor="#000000"
        Command="{Binding PushPage}"
        CommandParameter="634c5764-5f7b-4169-a084-6793c2a930fd"
        CornerRadius="8"
        WidthRequest="200"
        HeightRequest="40"
        FontAttributes="Bold"
        FontFamily="Segoe UI"
        Grid.Row="0"
        Grid.Column="1"
        Margin="10,25,20,0" />

         <Button
        x:Name="LeaderToolboxButton"
        Text="Leader Toolbox"
        StyleClass="btn, btn-primary"
        HorizontalOptions="Center"
        BackgroundColor="#7bb446"
        Command="{Binding PushPage}"
        CommandParameter="a9342a39-67f2-4ae2-b5d8-4ddc324c91e5"
        CornerRadius="8"
        WidthRequest="300"  HeightRequest="60"   FontSize="18"     FontAttributes="Bold"
        FontFamily="Segoe UI"
        Grid.Row="1"
        Grid.ColumnSpan="2" />

</Grid>
```

Explanation:

1. **<Grid>**: This is the layout container that organizes its children in rows and columns.
2. **<Grid.RowDefinitions>**: It defines two rows, the first one set to "Auto" height, and the second one also set to "Auto" height.
3. **<Grid.ColumnDefinitions>**: It defines two columns, both set to take up equal width ("*").
4. **<Button x:Name="MyScheduleButton" ...>**: This is the first **Button** named "MyScheduleButton." It has various properties set, such as text, style class, background color, command binding, corner radius, width, height, and margin. It is placed in the first row and the first column of the grid.
5. **<Button x:Name="GroupChatButton" ...>**: This is the second **Button** named "GroupChatButton." Similar to the first button, it has various properties set and is placed in the first row and the second column of the grid.
6. **<Button x:Name="LeaderToolboxButton" ...>**: This is the third **Button** named "LeaderToolboxButton." It has properties similar to the other buttons but spans both columns in the second row of the grid.

Overall, the code represents a grid layout with three buttons arranged in rows and columns. Each button has specific properties such as text, style, and commands associated with them.

The below code and explanation are of block Content:

```
<StackLayout Spacing="16"
  StyleClass="first-section-padding"
  Margin="0, 24, 0, 0">


  <StackLayout Spacing="0">


    <Grid ColumnDefinitions="*, 24"
      ColumnSpacing="12"
      Padding="0, 16">


      <ContentView Grid.Column="0"
        VerticalOptions="Center">
        <Label Text=" Resource Title  "
          StyleClass="h4-lowercase"
          FontAttributes="Bold" />
    </ContentView>

    <Rock:Icon IconClass="chevron-right"
        IconFamily="FontAwesomeSolid"
        TextColor="Black"
        FontSize="14"
        Grid.Column="1"
        VerticalOptions="Center"
        HorizontalOptions="Center" />

    <Grid.GestureRecognizers>
        <TapGestureRecognizer Command="{Binding OpenBrowser}"
            CommandParameter="http://gracetest.somee.com/" />
    </Grid.GestureRecognizers>
</Grid>

<Rock:Divider />

<Grid ColumnDefinitions="*, 24"
    ColumnSpacing="12"
    Padding="0, 16">

    <ContentView Grid.Column="0"
        VerticalOptions="Center">
        <Label Text=" Resource Title "
            StyleClass="h4-lowercase"
            FontAttributes="Bold"/>
    </ContentView>

    <Rock:Icon IconClass="chevron-right"
        IconFamily="FontAwesomeSolid"
        TextColor="Black"
        FontSize="14"
        Grid.Column="1"
        VerticalOptions="Center"
        HorizontalOptions="Center" />

    <Grid.GestureRecognizers>
        <TapGestureRecognizer Command="{Binding OpenBrowser}"
            CommandParameter="http://gracetest.somee.com/" />
    </Grid.GestureRecognizers>
</Grid>

<Rock:Divider />

<Grid ColumnDefinitions="*, 24"
    ColumnSpacing="12"
    Padding="0, 16">

    <ContentView Grid.Column="0"
        VerticalOptions="Center">
        <Label Text=" Resource Title "
            StyleClass="h4-lowercase"
            FontAttributes="Bold"/>
    </ContentView>

    <Rock:Icon IconClass="chevron-right"
        IconFamily="FontAwesomeSolid"
        TextColor="Black"
        FontSize="14"
        Grid.Column="1"
        VerticalOptions="Center"
        HorizontalOptions="Center" />

    <Grid.GestureRecognizers>
        <TapGestureRecognizer Command="{Binding OpenBrowser}"
            CommandParameter="http://gracetest.somee.com/" />
    </Grid.GestureRecognizers>
</Grid>
        <Rock:Divider />

<Grid ColumnDefinitions="*, 24"
    ColumnSpacing="12"
    Padding="0, 16">

    <ContentView Grid.Column="0"
        VerticalOptions="Center">
        <Label Text=" Resource Title "
            StyleClass="h4-lowercase"
            FontAttributes="Bold"/>
    </ContentView>

    <Rock:Icon IconClass="chevron-right"
        IconFamily="FontAwesomeSolid"
        TextColor="Black"
        FontSize="14"
        Grid.Column="1"
        VerticalOptions="Center"
        HorizontalOptions="Center" />

    <Grid.GestureRecognizers>
        <TapGestureRecognizer Command="{Binding OpenBrowser}"
            CommandParameter="http://gracetest.somee.com/" />
        </Grid.GestureRecognizers>
    </Grid>

    <Rock:Divider />

     <Grid ColumnDefinitions="*, 24"
        ColumnSpacing="12"
        Padding="0, 16">

        <ContentView Grid.Column="0"
            VerticalOptions="Center">
            <Label Text=" Resource Title "
                StyleClass="h4-lowercase"
                FontAttributes="Bold"/>
        </ContentView>

        <Rock:Icon IconClass="chevron-right"
            IconFamily="FontAwesomeSolid"
            TextColor="Black"
            FontSize="14"
            Grid.Column="1"
            VerticalOptions="Center"
            HorizontalOptions="Center" />

        <Grid.GestureRecognizers>
            <TapGestureRecognizer Command="{Binding OpenBrowser}"
                CommandParameter="http://gracetest.somee.com/" />
        </Grid.GestureRecognizers>
    </Grid>

</StackLayout>
```

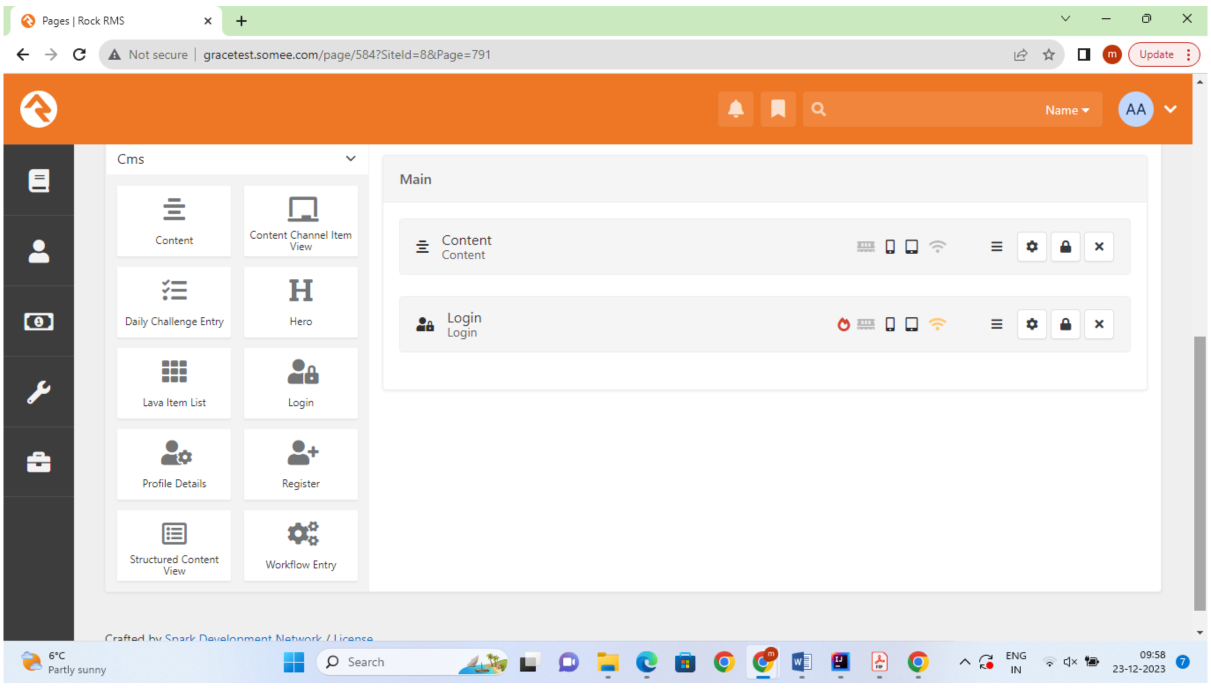
</StackLayout>

Explanation:

1. **<StackLayout>**: This is the main container with spacing, style class, and margin properties.
2. **<Grid ColumnDefinitions="*, 24" ...>**: Each **Grid** represents an item in the list. It has two columns, the first one taking up the available space (*****), and the second one with a fixed width of 24 units. Padding and column spacing are also set.
3. **<ContentView Grid.Column="0" ...>**: The content view within the first column of the grid. It contains a **Label** displaying the resource title.
4. **<Rock:Icon>**: This is a custom control (**Rock:Icon**) representing a chevron-right icon. It is placed in the second column of the grid.
5. **<Grid.GestureRecognizers>**: This section contains a **TapGestureRecognizer** that triggers the **OpenBrowser** command with a specific URL parameter when the user taps on the grid.
6. **<Rock:Divider />**: This is a custom control (**Rock:Divider**) representing a visual divider or separator between items in the list.
7. The entire structure is repeated for each item in the list.

The code essentially creates a vertical list of items, each consisting of a title, a chevron-right icon, and a divider. Tapping on each item triggers the **OpenBrowser** command with a specific URL parameter

<u>Login:</u>



The below code and explanation are of block Content:

  <Rock:Image

```
   Source="https://i.ibb.co/hM0W1G9/download-1.jpg"
   BackgroundColor="#c4c4c4"
   HeightRequest="128"
   WidthRequest="128"
   Aspect="AspectFill"
    Margin="0,70,0,30"
   HorizontalOptions="Center"
   VerticalOptions="Fill">
   <Rock:CircleTransformation BorderSize="4"
       BorderColor="rgba(255, 255, 255, 0.58)" />
</Rock:Image>
```

Explanation:

<Rock:Image>: This is a custom image control (Rock:Image). It displays an image from the specified URL.

Source: The URL of the image to be displayed (https://i.ibb.co/hM0W1G9/download-1.jpg).

BackgroundColor: The background color of the image (in case the aspect ratio results in empty spaces).

HeightRequest and WidthRequest: The requested height and width of the image.

Aspect="AspectFill": Specifies how the image should be resized to fill its container while maintaining its aspect ratio.

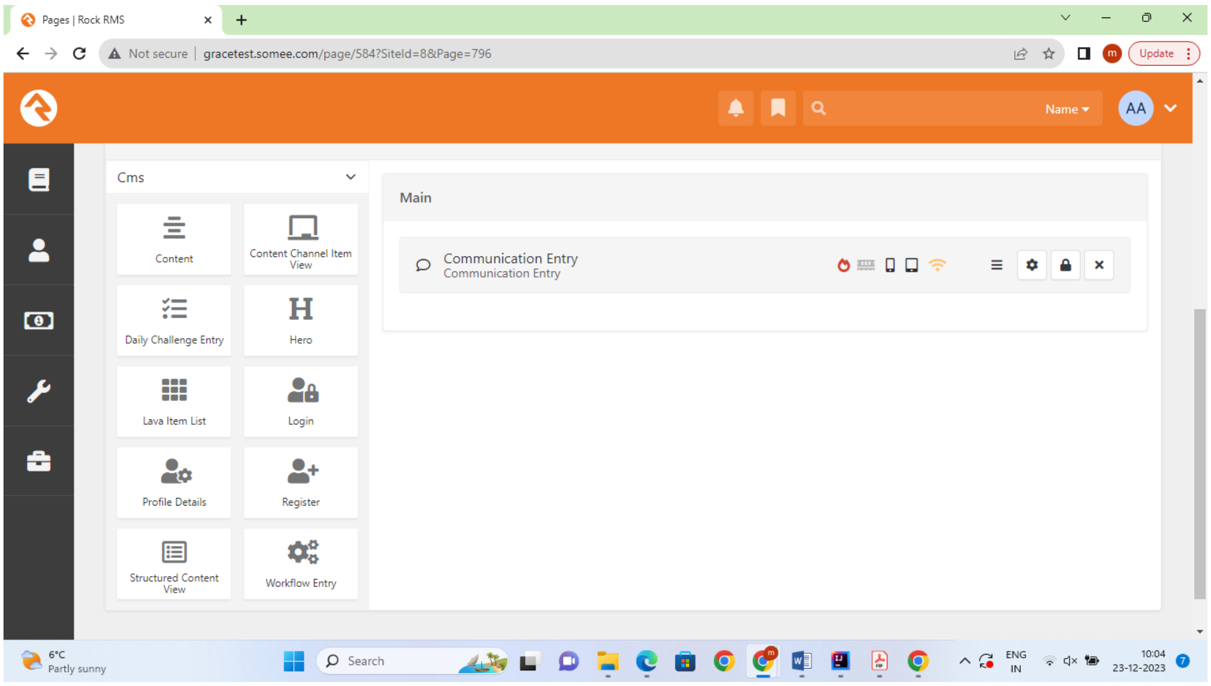
<Rock:CircleTransformation>: This is a custom transformation applied to the Rock:Image, creating a circular effect.

BorderSize="4": Specifies the size of the border around the circular image.

BorderColor="rgba(255, 255, 255, 0.58)": Specifies the color of the border in RGBA format (red, green, blue, alpha), creating a semi-transparent white border.

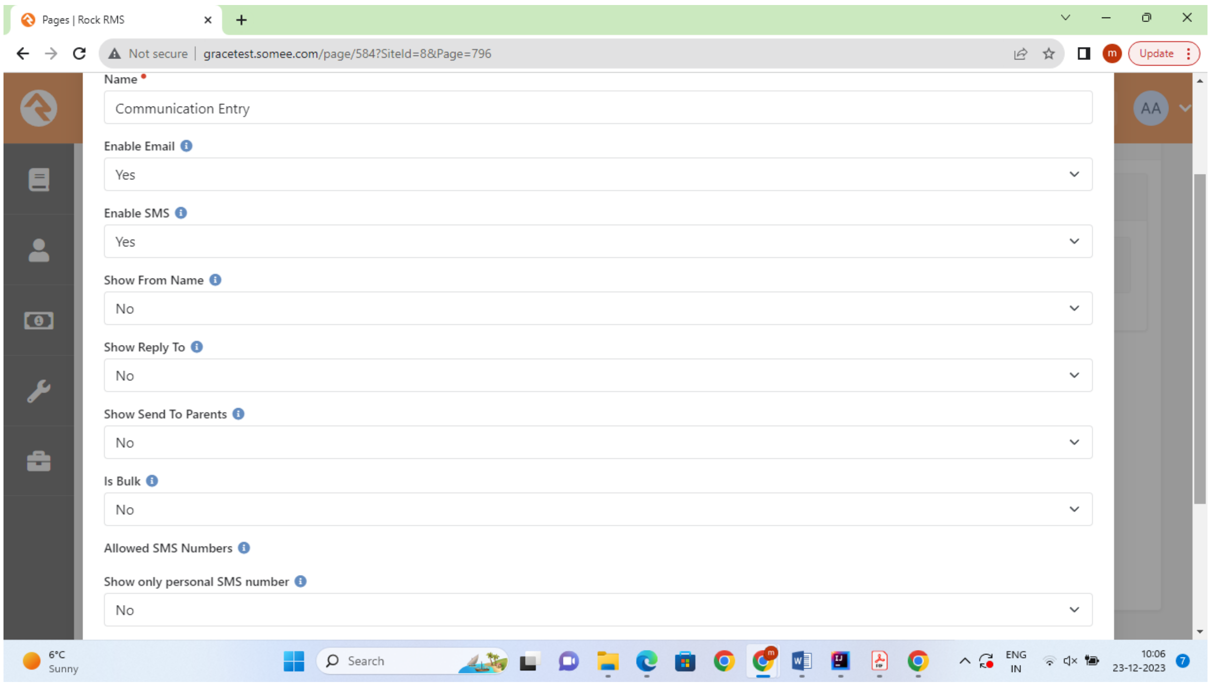
The Rock:Image control is positioned in the layout with specific margin and alignment settings (HorizontalOptions="Center" and VerticalOptions="Fill"). The circular transformation adds a border to the image, giving it a circular appearance with a white semi-transparent border.

Chat:



The below code and explanation are of block Communication Entry:

For Communication Entry Block used premade block and changed settings as shown in the below screenshot



Name *
Communication Entry

Enable Email
Yes

Enable SMS
Yes

Show From Name
No

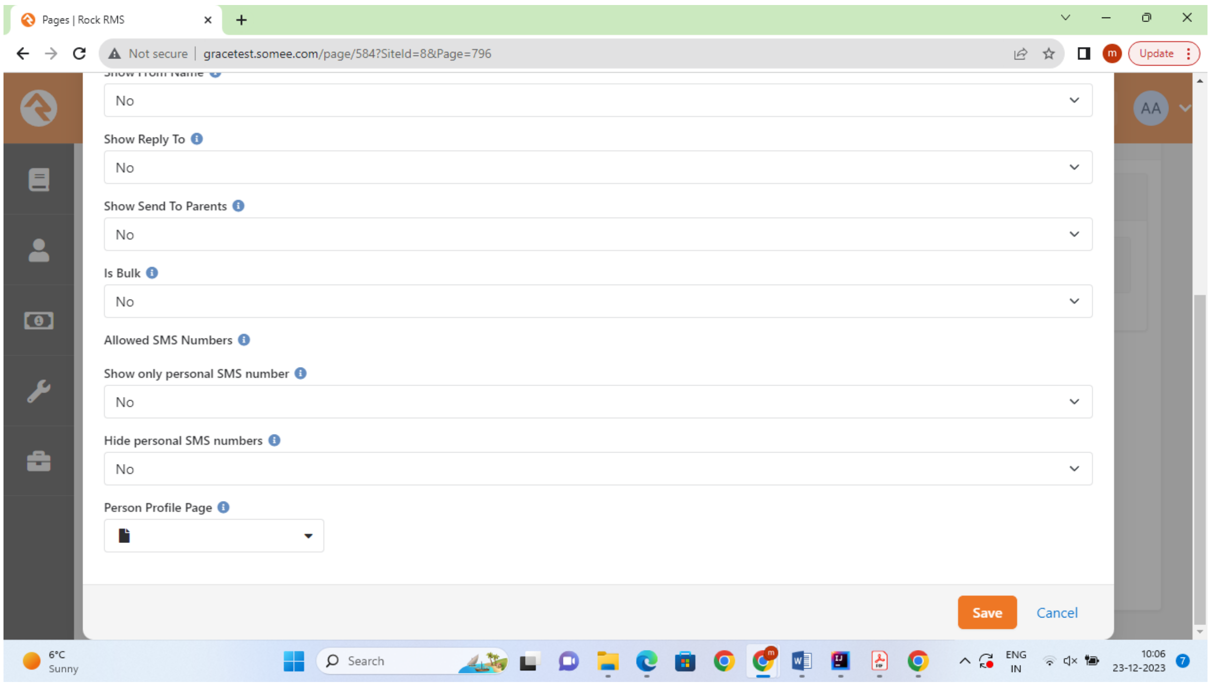
Show Reply To
No

Show Send To Parents
No

Is Bulk
No

Allowed SMS Numbers

Show only personal SMS number
No


Pages | Rock RMS

Not secure | gracetest.somee.com/page/584?SiteId=8&Page=796

No

Show Reply To
No

Show Send To Parents
No

Is Bulk
No

Allowed SMS Numbers

Show only personal SMS number
No

Hide personal SMS numbers
No

Person Profile Page

Save Cancel

6°C Sunny

10:06 23-12-2023