Project work 1

Summary of the Issue

The issue addressed in this project is the need for a software tool that enables UNDAC Team Leaders to manage team alerts efficiently. Team alerts can be created by any team member, and they should include information such as the alert's title, status, and description. These alerts need to be displayed, filtered, and new alerts created, all within a user-friendly interface.

Snippets from My Code with Commentary

The `TeamAlert` class is defined to structure alert data. It encapsulates alert properties, such as `Title`, `Status`, and `Description`, following good software design practice.

```
using SQLite;

namespace Undac.Models
{
    public class TeamAlert
    {
        public string Title { get; set; }
        public string Status { get; set; }
        public string Description { get; set; }
    }
}
```

We can see the code here :

Link to my TeamAlert.cs

My XAML page contains several essential elements for team alert page.

To begin, I use the `<Grid>` tag, which allows me to structure the layout of the page by specifying rows and columns.

At the heart of this page is the `<ListView>`, which I've named "alertListView." This component is crucial as it displays the list of team alerts, allowing for a quick overview of information regarding each alert.

To interact with the alerts, I've integrated buttons using the `<Button>` tag. These buttons serve specific purposes: "Filter" to narrow down the list based on the alert status, "Clear Filter" to reset the list, and "Add Alert" to create new alerts. They facilitate alert management by providing direct actions to the user.

Lastly, the `<Entry>` tag completes the interface by offering a text field where the user can input an alert status. This enables quick and precise filtering of alerts based on the desired status.

```
    <Grid>

        <ListView x:Name="alertListView">
            <ListView.ItemTemplate>

                //Code

            </ListView.ItemTemplate>
        </ListView>


       <Entry x:Name="filterEntry" Placeholder="Filter by Status"
HorizontalOptions="Start" VerticalOptions="End" Margin="0,0,20,20"/>
       <Button Text="Filter" Clicked="FilterButton_Clicked"
HorizontalOptions="Start" VerticalOptions="End" Margin="0,25,20,20"/>
       <Button Text="Clear Filter" Clicked="ClearFilterButton_Clicked"
HorizontalOptions="Start" VerticalOptions="End" Margin="0,50,20,20"/>
       <Button Text="Create Alert" Clicked="CreateAlertButton_Clicked"
HorizontalOptions="End" VerticalOptions="End" Margin="0,0,20,20" />


    </Grid>
```

We can see the all xaml code here :

Link to my AlertPage.xaml

This page uses an observable collection called `"alerts"` to store and manage alerts. Upon page initialization, the list of alerts is displayed in a ListView named `"alertListView"`. Three event handlers are associated with buttons: `"Filter"` to filter alerts based on a provided status, `"Clear Filter"` to reset the list, and `"Create Alert"` to enable users to create new alerts by entering a title, selecting a status, and adding a description. To be able to write these codes, I used programming websites for assistance with C# ; particularly regarding the `Display` functions.

```
using System;
using System.Collections.Generic;
using System.Collections.ObjectModel;
using Microsoft.Maui.Controls;
using Undac.Models;

namespace Undac
{
    public partial class AlertPage : ContentPage
    {
        private ObservableCollection<TeamAlert> alerts = new
ObservableCollection<TeamAlert>();
```

```
        public AlertPage()
        {
            //Code...
        }

        private void FilterButton_Clicked(object AlertSender, EventArgs e)
        {
           //Code...
        }

        private void ClearFilterButton_Clicked(object AlertSender, EventArgs e)
        {
            //Code...
        }

        private async void CreateAlertButton_Clicked(object AlertSender, EventArgs
e)
        {

            //Code...
        }

    }
}
```

We can see the c# code here :

Link to my AlertPage.xaml.cs

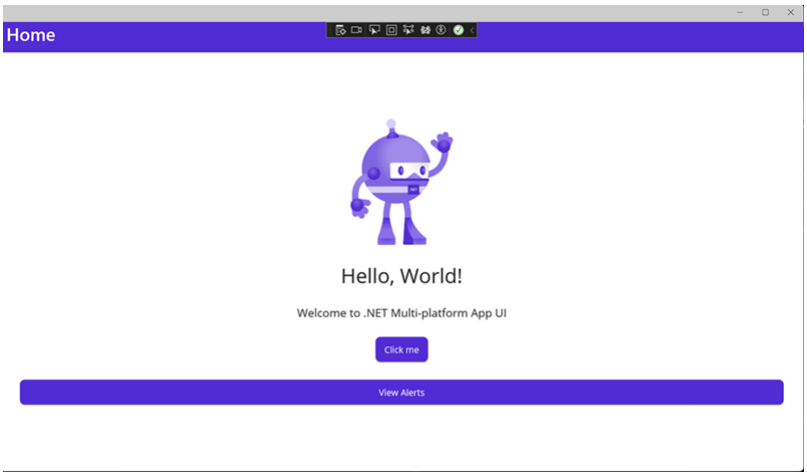
General Reflective Section

Overall, the task is quite successful, and I have been able to learn a lot about coding in C#. I applied what we have learned in the course so far to create simple, clear, and readable code. It particularly adheres to what was covered in week 5, which is documentation.

The code is being reviewed by other team members. However, due to an issue with the database, the code is not yet complete. But we are working together with the team to resolve this issue. The code has also not been tested yet.

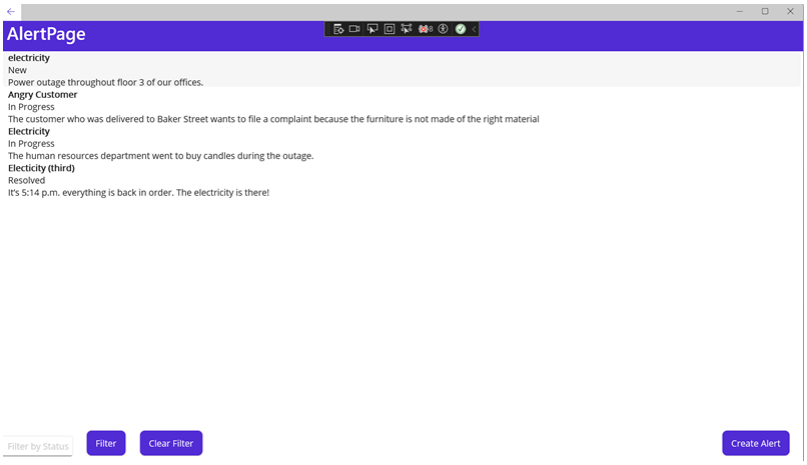
Some screenshots about the page :

-



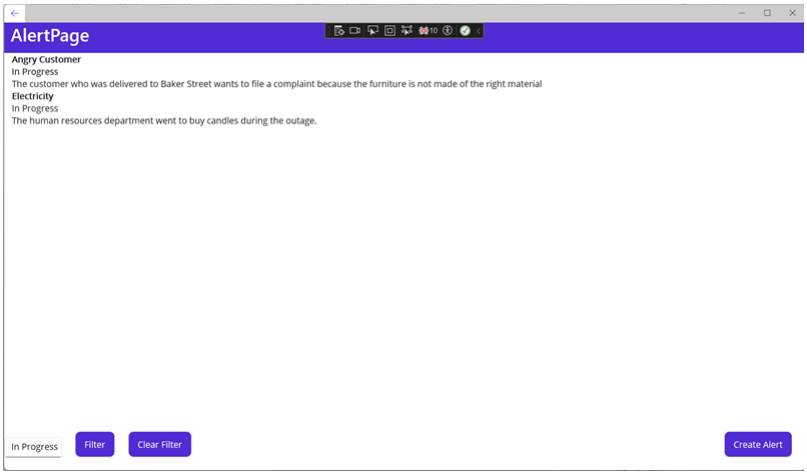
Here, we can see the ViewAlert button which directs to the alerts page.

-



Here, it's the alert page with some alerts I created with the button "Create Alert"

-



Here, I used the status filter by typing 'In progress,' and we can see the two alerts that have the status 'In progress.'