University of Zakho

Faculty of Science

Department of Computer Science

Mobile Application

Group (A)

Report

(Pharmacy Application)

Prepared By

Said Zewar

Brisk Kamil

Mohamad Jamil

Mahmod Shawkat

Supervisor

Yousif Garabet Arshak

Table of Contents

Chapter1

Introduction

Mobile apps, like web apps, are on the rise today. Let’s think about why these applications are so popular for both users and businesses. It's really simple.

If the user installs our application, we can inform them about news, benefits, etc. easily and directly on their phone and thus increase the chance to sell something. This would probably be difficult to achieve with an email newsletter, which the user usually deletes straightaway without reading it at all. As a bonus, we'll get information about the user's device, such as their location, so we can target our offers to a specific city / location and adjust the product offer accordingly. That was for businesses. And for the mobile users, the benefits of mobile apps are that they can browse most of the content offline,

If you’re the owner of a pharmacy, you probably want to know how to attract more customers. The task isn't really easy, because there're a lot of drugstores in the healthcare market, and each one does its best to interest a maximum of clients. A good solution in such a case is to use advanced software, namely, an online pharmacy.

We're talking about special "online offices" of pharmacy companies: mobile and web platforms that allow interacting with drugstore customers remotely (which is especially important after the 2020 pandemic). Users of these apps solve a lot of problems associated with the need to go to the pharmacy; and you, as the owner of the software in question, improve your service quality.

Do you want to know more? Then read our article to find out 7 reasons to start pharmacy application development.

The Benefits of Having a Mobile App in Pharmacy

As promised: here are the 7 main benefits of having a pharmacy online. Read them carefully.

1.Digital marketing platform

Medical mobile app development will allow you to get an innovative tool for promoting your pharmacy service. And you’re free to use it with convenience, as and when you want! Moreover, you'll be able to find out what your customers really like and dislike when it comes to your drugstore. As a result, you'll be constantly perfecting your product (which leads to increased customer loyalty).

2. Gaining more customers

The real chance to communicate with your customers is a huge advantage. You don't just meet their needs and satisfy their desires, you become closer and clearer to them, you are always "at hand." Use your pharmacy app to the maximum by interacting with the client through push notifications and timely informing him about discounts and promotions... in other words, turn users into loyal customers.

3. Improved competitive ability

As you might have already guessed yourself, pharmacy apps are just perfect for growing the competitiveness of the company they belong to. There is nothing surprising since these services help you communicate with customers in real-time, when and where you need to.

4. Recognizable corporate brand

It won't be superfluous to create a recognizable pharmacy brand to embody the company's image. And the mobile app with an appealing UI design might help to implement the idea by becoming a logical continuation of the pharmacy chain brand.

5. Online sales of drugs

Online drug sales are another reason to initiate healthcare mobile app development. Roughly speaking, we're facing a classic win-win situation where both parties to the process are satisfied:

consumers are used to buying almost everything online (especially during the quarantine and forced self-isolation);

by satisfying one more request of your consumer, you increase the level of sales and, accordingly, income.

6. Non-stop business development

Moving is living, as they say. And this rule applies, among other things, to the pharmacy business. To keep your company afloat, you need to meet (or even anticipate) the expectations of your customers. And the pharmacy drug app is a great tool to understand your target audience.

What you should do is add analytical features to the application and study user behavior. With such a smart approach, you'll receive a lot of useful information to continue to grow your business.

7. Improved patient care

Of course, a drugstore is hardly a hospital, and you cannot really treat your customers. However, pharmacy apps are still able to provide some help to patients. How about background materials on various medications and their health effects? Or why not add an online chat to allow users to consult with your experts?

Though, we'll discuss your opportunities later when describing pharmacy app features at length.

There are currently 3 platforms (Android, iOS, and Windows) and each uses a different programming language.

- Android (Java, Kotlin)
- iOS (Swift)
- Windows (C#)

So if our customer wants to create an application for all these platforms, it'd be a hell of work. If you ever wondered it's unnecessary to write the same application 3 times - one for Android, the other for iOS, the third for Windows, and each in a different language, then you're in the right place.

What is Xamarin?

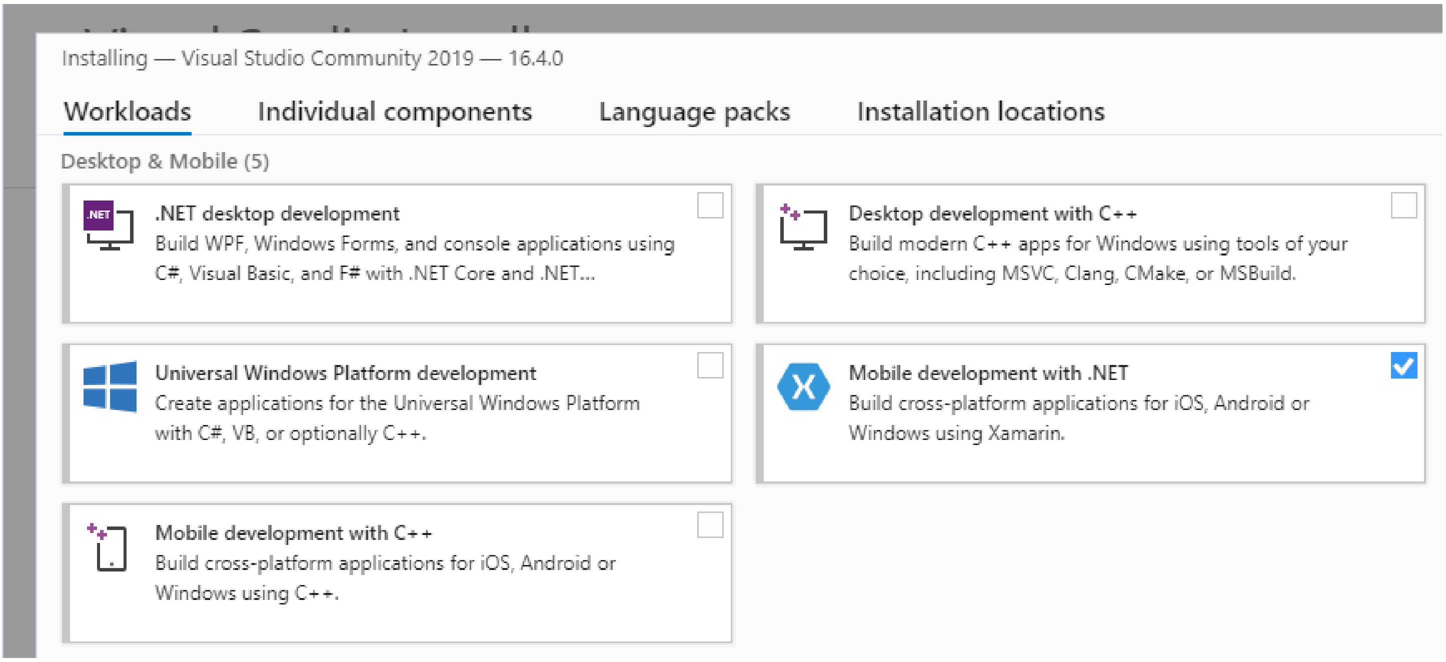
Xamarin allows to develop native applications for all the platforms at once, with a minimum of OS-dependent scripts and all using only C# .NET and XAML. So we can share about 90% of common code across all platforms. Haven't you heard of XAML yet? Never mind. As the name suggests, it's a classic XML with extended syntax (eXtensible Application Markup Language). We'll use it to design the visual layout of our applications, which we'll then "bring to life" with C# scripts.

And how does it actually work? As I mentioned earlier, it's kinda revolutionary that Xamarin development is native and in C# .NET at the same time, even for platforms with different native languages. The principle is again very simple. When we build the project, it's compiled to the native subsystems (Java for Android, Swift for iOS) and each application then looks and behaves as if it was written in the native language for the platform. Certainly now we all see the major advantages of Xamarin. Another indisputable advantage is that if we don't know Java or Swift and want to create even an application just for Android or iOS, we can do it in C#. Xamarin offers 5 basic solution types: Xamarin.Forms (development for all platforms at once) Android application
iOS application Android Wear application WatchOS application

Chapter 2

Background

1-Xamarin.forms



Installing — Visual Studio Community 2019 — 16.4.0

Desktop & Mobile (5)

□ .NET desktop development
Build WPF, Windows Forms, and console applications using C#, Visual Basic, and F# with .NET Core and .NET...

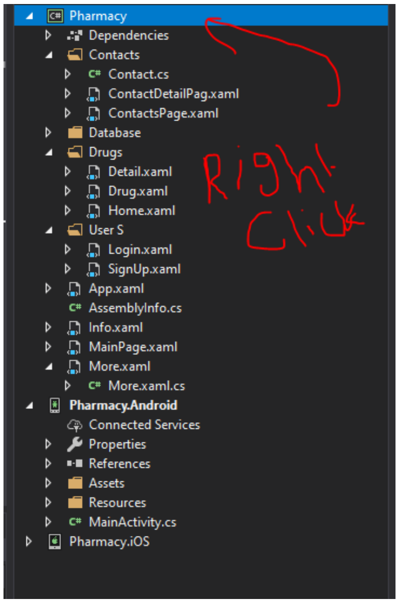
□ Desktop development with C++
Build modern C++ apps for Windows using tools of your choice, including MSVC, Clang, CMake, or MSBuild.

□ Universal Windows Platform development
Create applications for the Universal Windows Platform with C#, VB, or optionally C++.

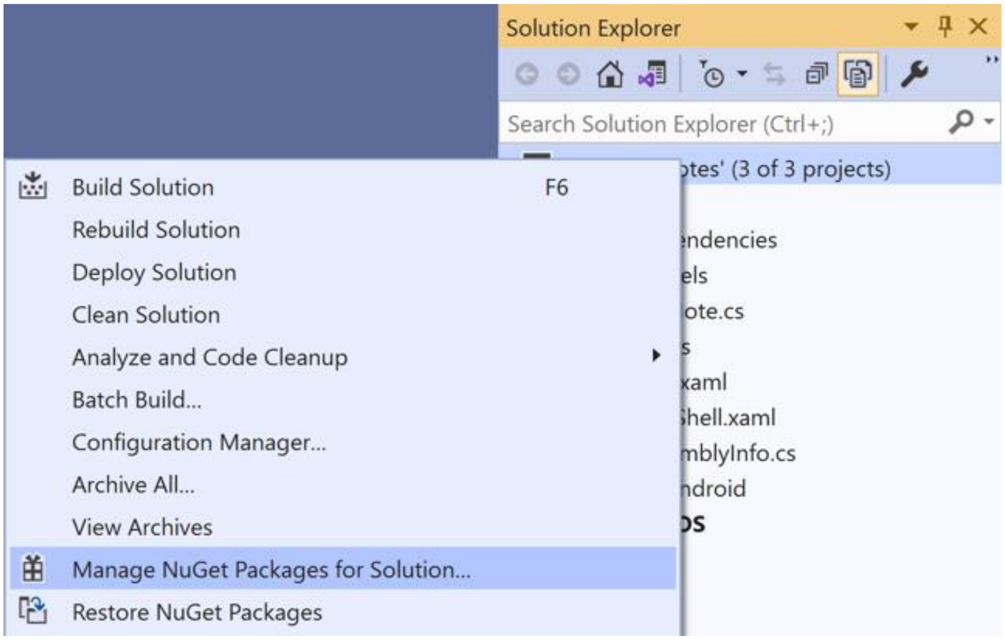
☑ Mobile development with .NET
Build cross-platform applications for iOS, Android or Windows using Xamarin.

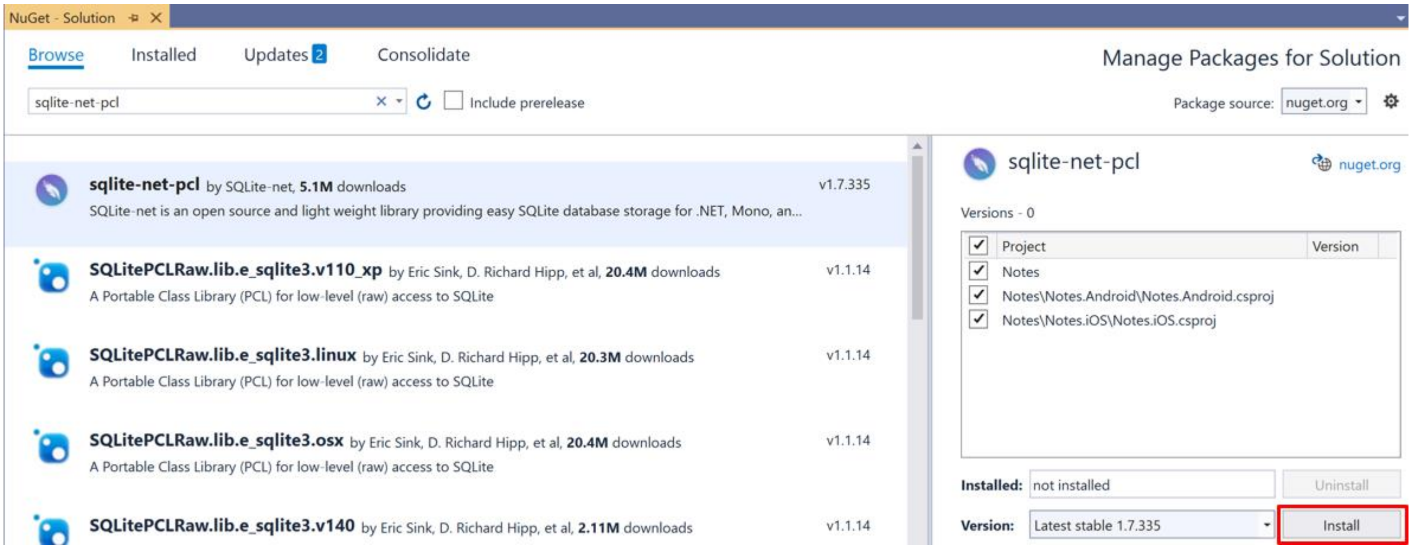
□ Mobile development with C++
Build cross-platform applications for iOS, Android or Windows using C++.

2-SqlLite



```
Pharmacy
  Dependencies
  Contacts
    Contact.cs
    ContactDetailPag.xaml
    ContactsPage.xaml
  Database
  Drugs
    Detail.xaml
    Drug.xaml
    Home.xaml
  User S
    Login.xaml
    SignUp.xaml
  App.xaml
  AssemblyInfo.cs
  Info.xaml
  MainPage.xaml
  More.xaml
    More.xaml.cs
Pharmacy.Android
  Connected Services
  Properties
  References
  Assets
  Resources
  MainActivity.cs
Pharmacy.iOS
```





sqlite-net-pcl

sqlite-net-pcl by SQLite-net, **5.1M** downloads v1.7.335
SQLite-net is an open source and light weight library providing easy SQLite database storage for .NET, Mono, an...

SQLitePCLRaw.lib.e_sqlite3.v110_xp by Eric Sink, D. Richard Hipp, et al, **20.4M** downloads v1.1.14
A Portable Class Library (PCL) for low-level (raw) access to SQLite

SQLitePCLRaw.lib.e_sqlite3.linux by Eric Sink, D. Richard Hipp, et al, **20.3M** downloads v1.1.14
A Portable Class Library (PCL) for low-level (raw) access to SQLite

SQLitePCLRaw.lib.e_sqlite3.osx by Eric Sink, D. Richard Hipp, et al, **20.4M** downloads v1.1.14
A Portable Class Library (PCL) for low-level (raw) access to SQLite

SQLitePCLRaw.lib.e_sqlite3.v140 by Eric Sink, D. Richard Hipp, et al, **2.11M** downloads v1.1.14

sqlite-net-pcl

Versions - 0

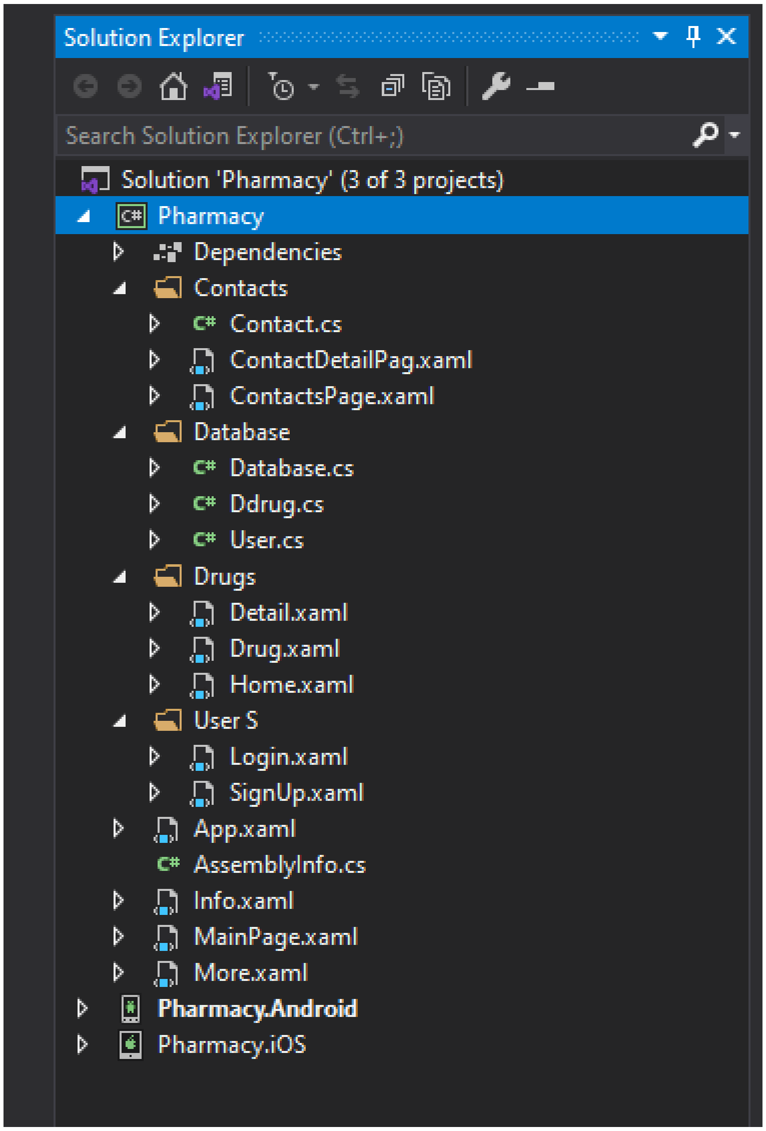
☑	Project	Version
☑	Notes	
☑	Notes\Notes.Android\Notes.Android.csproj	
☑	Notes\Notes.iOS\Notes.iOS.csproj	

Installed: not installed

Version: Latest stable 1.7.335

Chapter3

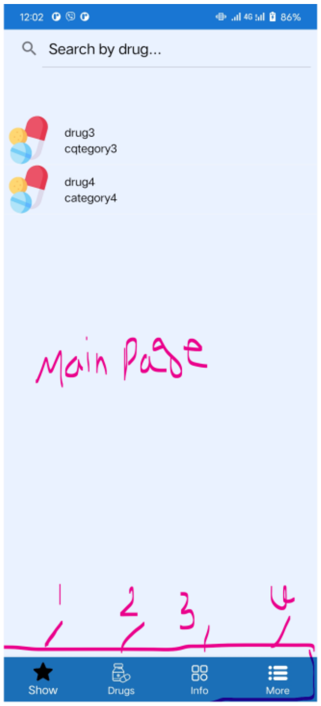
Application Implementation



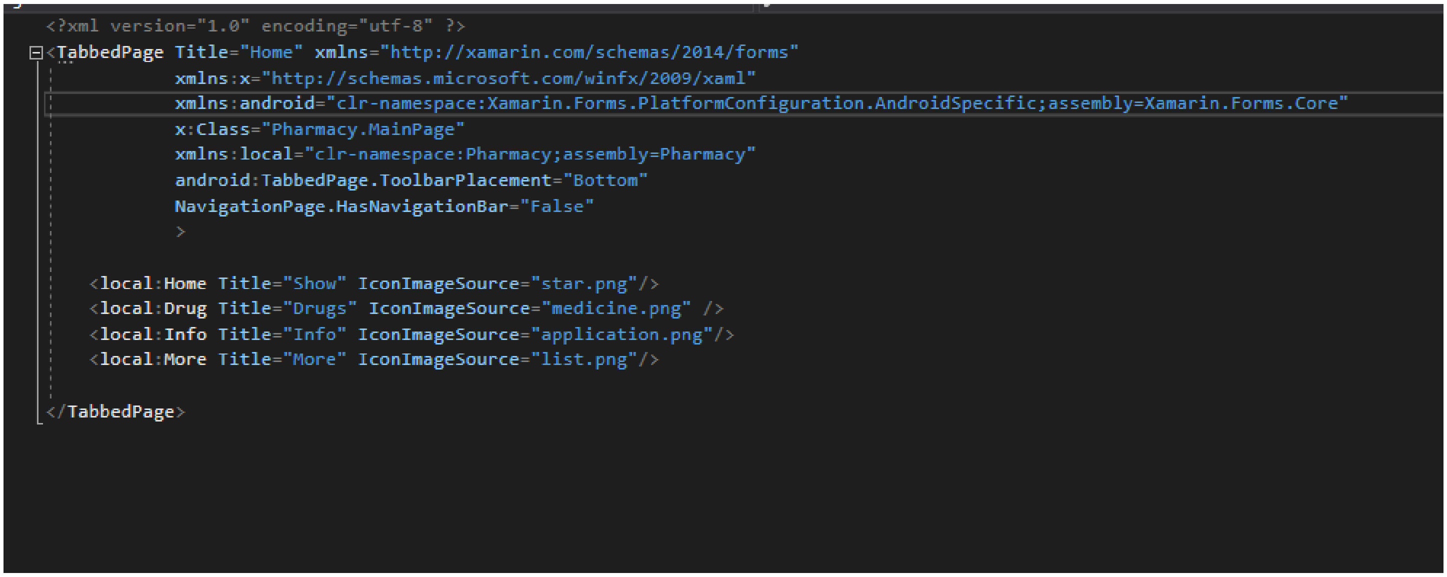
```
Solution 'Pharmacy' (3 of 3 projects)
  Pharmacy
    Dependencies
    Contacts
      Contact.cs
      ContactDetailPag.xaml
      ContactsPage.xaml
    Database
      Database.cs
      Ddrug.cs
      User.cs
    Drugs
      Detail.xaml
      Drug.xaml
      Home.xaml
    User S
      Login.xaml
      SignUp.xaml
    App.xaml
    AssemblyInfo.cs
    Info.xaml
    MainPage.xaml
    More.xaml
  Pharmacy.Android
  Pharmacy.iOS
```

Folders and Files

Application Design



Main Page

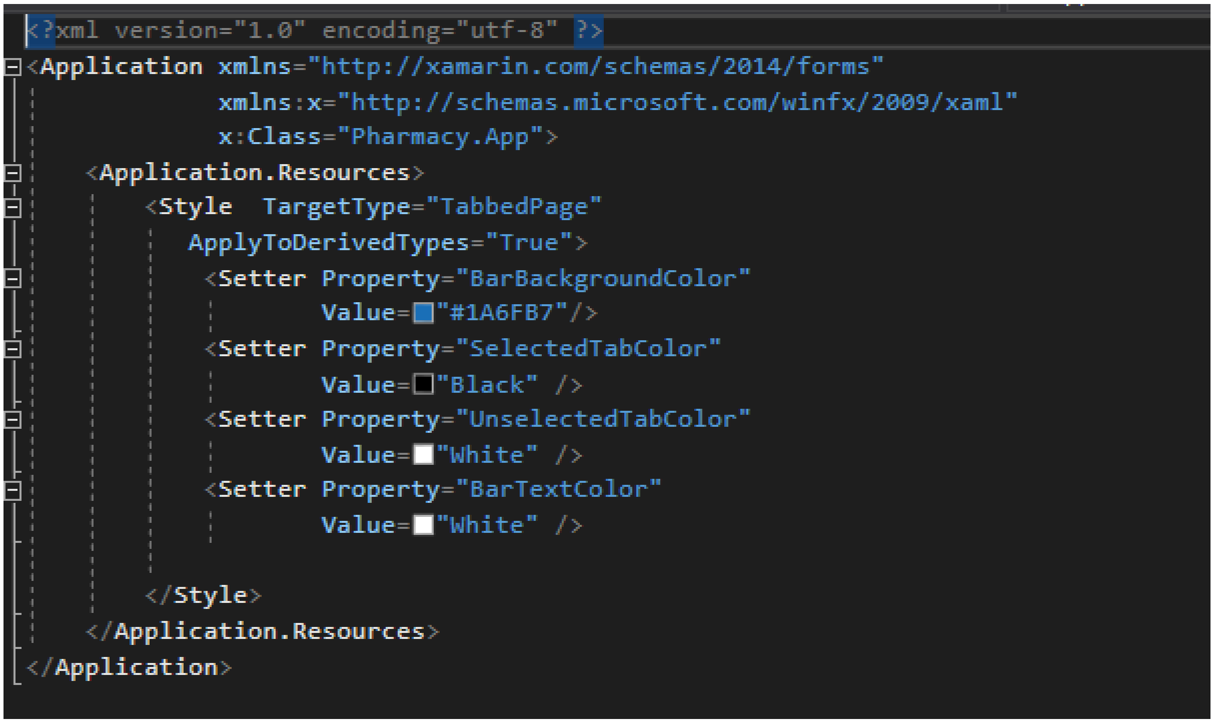


```
<?xml version="1.0" encoding="utf-8" ?>
<TabbedPage Title="Home" xmlns="http://xamarin.com/schemas/2014/forms"
            xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
            xmlns:android="clr-namespace:Xamarin.Forms.PlatformConfiguration.AndroidSpecific;assembly=Xamarin.Forms.Core"
            x:Class="Pharmacy.MainPage"
            xmlns:local="clr-namespace:Pharmacy;assembly=Pharmacy"
            android:TabbedPage.ToolbarPlacement="Bottom"
            NavigationPage.HasNavigationBar="False"
            >

    <local:Home Title="Show" IconImageSource="star.png"/>
    <local:Drug Title="Drugs" IconImageSource="medicine.png" />
    <local:Info Title="Info" IconImageSource="application.png"/>
    <local:More Title="More" IconImageSource="list.png"/>

</TabbedPage>
```

We used tabbed page as shown

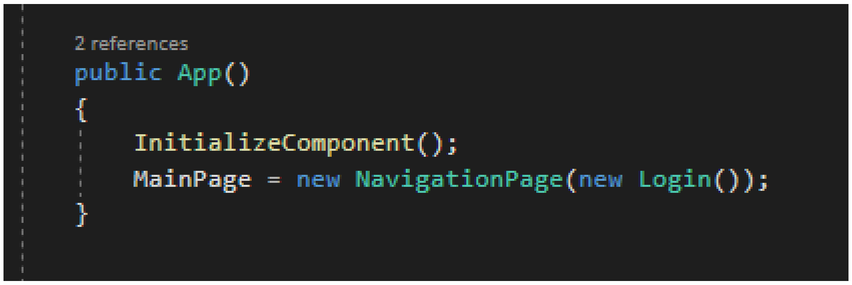
And in the ttabbed page we have (Home,Drug,Info,More)

App.xaml



```
<?xml version="1.0" encoding="utf-8" ?>
<Application xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="Pharmacy.App">
    <Application.Resources>
        <Style  TargetType="TabbedPage"
          ApplyToDerivedTypes="True">
            <Setter Property="BarBackgroundColor"
                    Value="#1A6FB7"/>
            <Setter Property="SelectedTabColor"
                    Value="Black" />
            <Setter Property="UnselectedTabColor"
                    Value="White" />
            <Setter Property="BarTextColor"
                    Value="White" />

        </Style>
    </Application.Resources>
</Application>
```

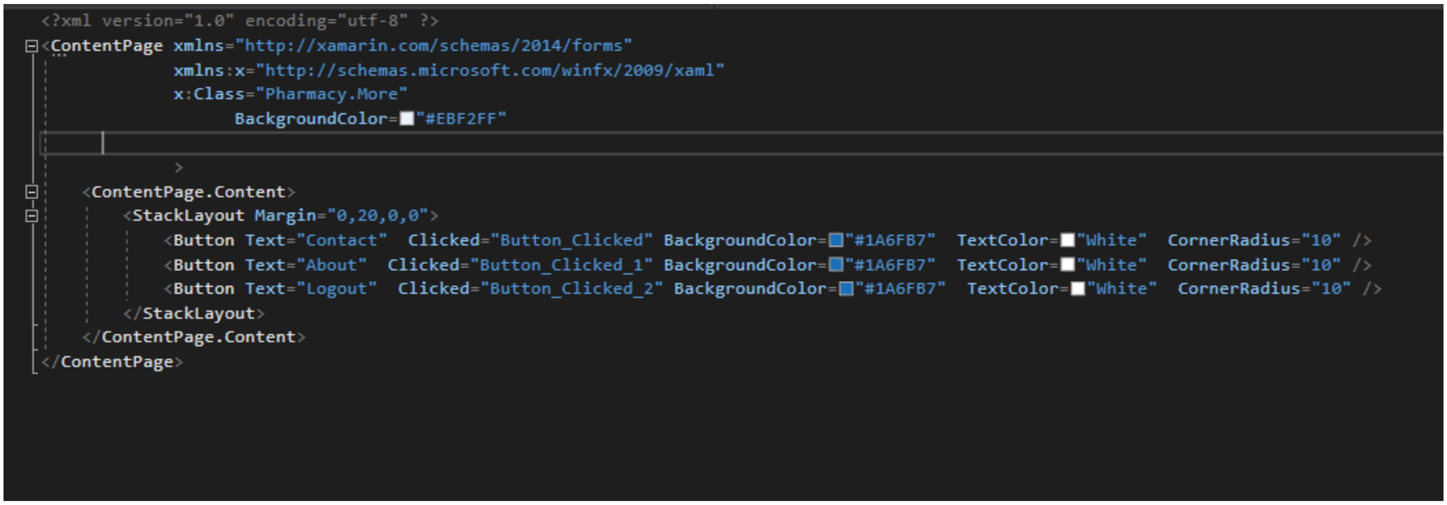
App.xaml.cs



```
2 references
public App()
{
    InitializeComponent();
    MainPage = new NavigationPage(new Login());
}
```

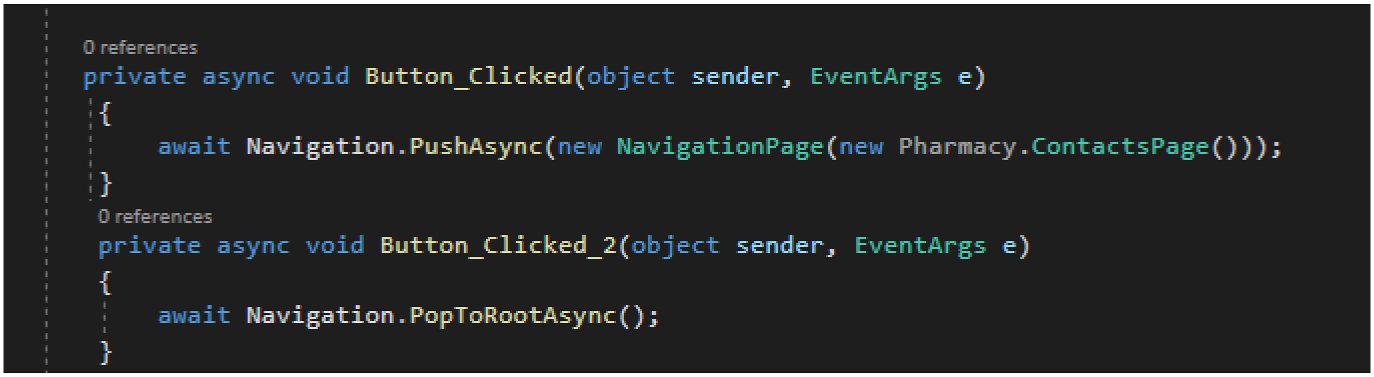
The Root is **Login Page** we will talk about it in the next pages.

More.xaml



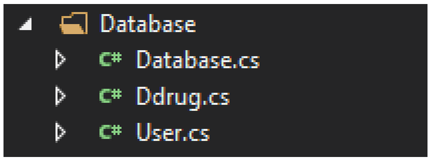
```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="Pharmacy.More"
                   BackgroundColor=□"#EBF2FF"
             
             >
    <ContentPage.Content>
        <StackLayout Margin="0,20,0,0">
            <Button Text="Contact"  Clicked="Button_Clicked" BackgroundColor=🟦"#1A6FB7"  TextColor=□"White"  CornerRadius="10" />
            <Button Text="About"  Clicked="Button_Clicked_1" BackgroundColor=🟦"#1A6FB7"  TextColor=□"White"  CornerRadius="10" />
            <Button Text="Logout"  Clicked="Button_Clicked_2" BackgroundColor=🟦"#1A6FB7"  TextColor=□"White"  CornerRadius="10" />
        </StackLayout>
    </ContentPage.Content>
</ContentPage>
```

More.xaml.cs



```
    0 references
    private async void Button_Clicked(object sender, EventArgs e)
    {
        await Navigation.PushAsync(new NavigationPage(new Pharmacy.ContactsPage()));
    }
     0 references
     private async void Button_Clicked_2(object sender, EventArgs e)
     {
        await Navigation.PopToRootAsync();
     }
```

Database design

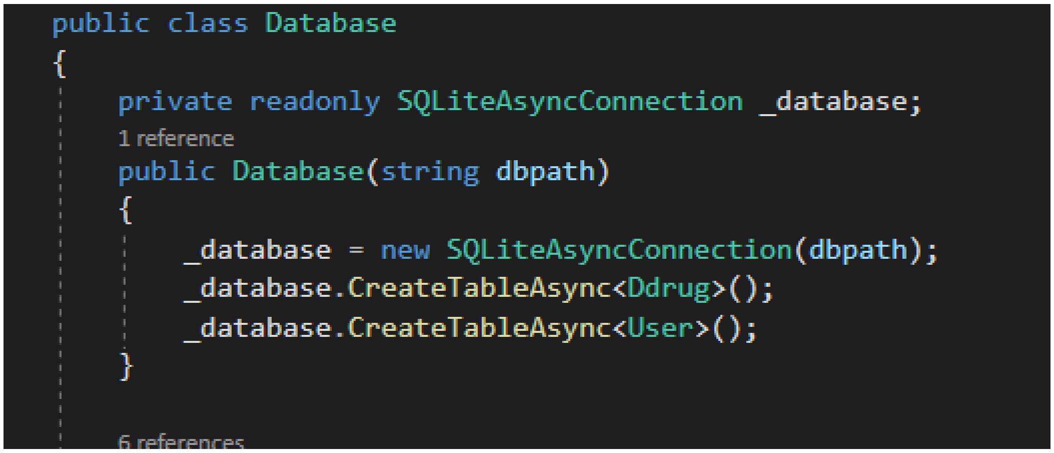


Store data in a local SQLite.NET database

We used local database to load and save **drugs item** into database and also to create a new **user** and **login system** (Login ,Signup) we will talk in details

Connection

Database.cs



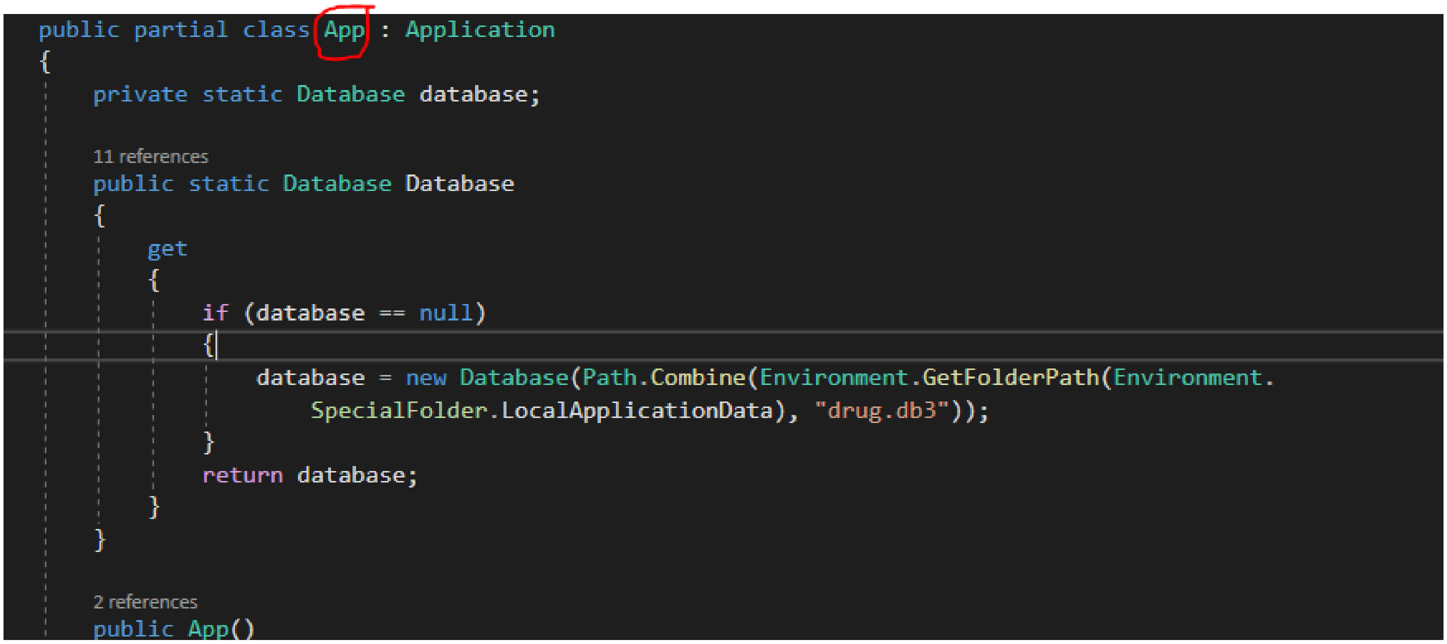
```
public class Database
{
    private readonly SQLiteAsyncConnection _database;
    1 reference
    public Database(string dbpath)
    {
        _database = new SQLiteAsyncConnection(dbpath);
        _database.CreateTableAsync<Ddrug>();
        _database.CreateTableAsync<User>();
    }

    6 references
```

Here we created a class with **database**.cs name and created two tables under Ddrug name And User name

As shown in the above image

When we want to deals with database we use class.cs and code first

App.xaml.css

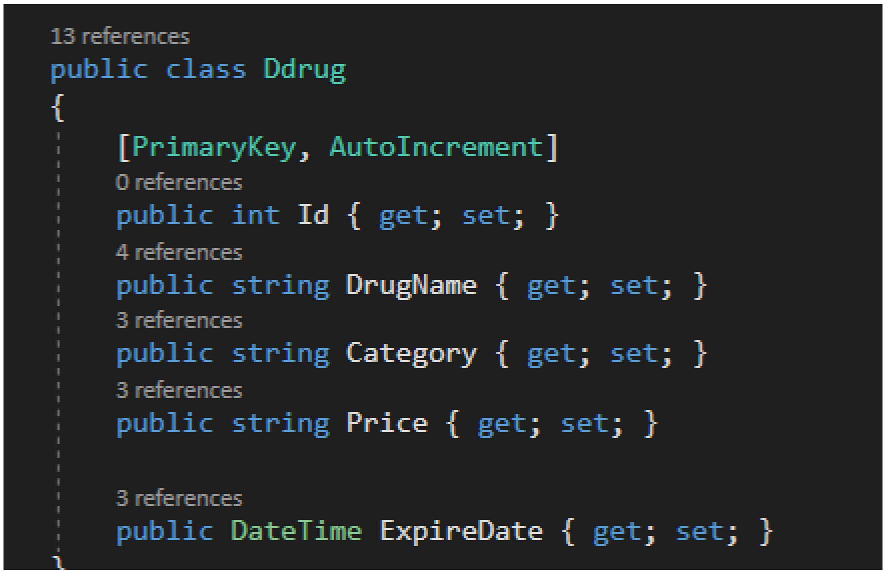


```
public partial class App : Application
{
    private static Database database;

    11 references
    public static Database Database
    {
        get
        {
            if (database == null)
            {
                database = new Database(Path.Combine(Environment.GetFolderPath(Environment.
                    SpecialFolder.LocalApplicationData), "drug.db3"));
            }
            return database;
        }
    }

    2 references
    public App()
```

We connected applicatioin with database

When we need store and show and delete and update data we should call App.Database. we will talk more in the next pages

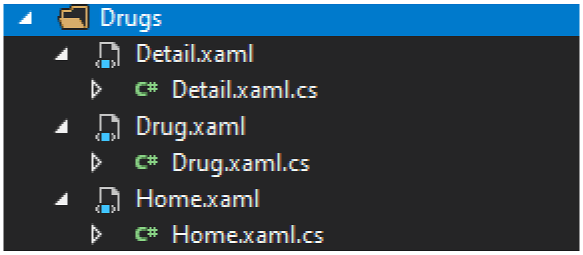
Ddrug.cs



```
13 references
public class Ddrug
{
    [PrimaryKey, AutoIncrement]
    0 references
    public int Id { get; set; }
    4 references
    public string DrugName { get; set; }
    3 references
    public string Category { get; set; }
    3 references
    public string Price { get; set; }

    3 references
    public DateTime ExpireDate { get; set; }
}
```

In this file we declare what the attributes (ID,DrugName,Category,ExpireDate) of the Ddrugs table, and what type they are, and we also can create constraints such as the id Primary key, and AutoIncrement Constraints.

Drug Folder

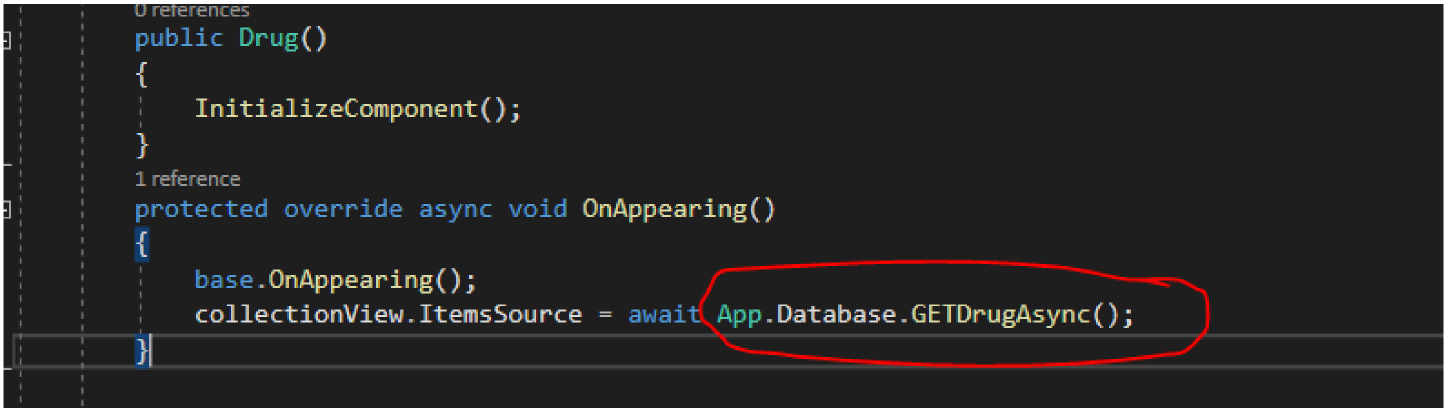


```
Drugs
    Detail.xaml
        Detail.xaml.cs
    Drug.xaml
        Drug.xaml.cs
    Home.xaml
        Home.xaml.cs
```

In this folder we have a Home.xaml Page and Drug.Xaml Page ad Detail xaml Page Each file contain a specific design and code

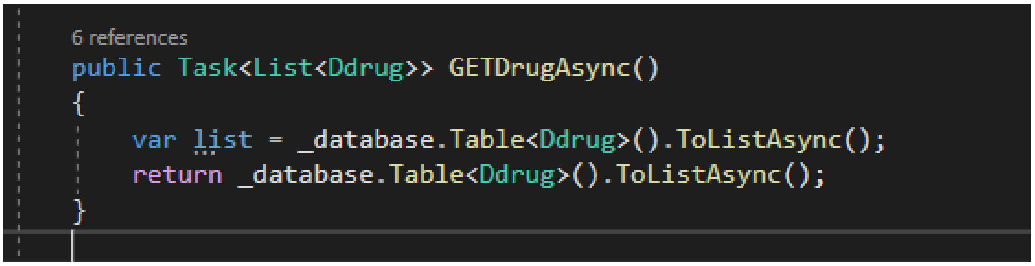
Drug.xaml.cs

This method to Load Data from database

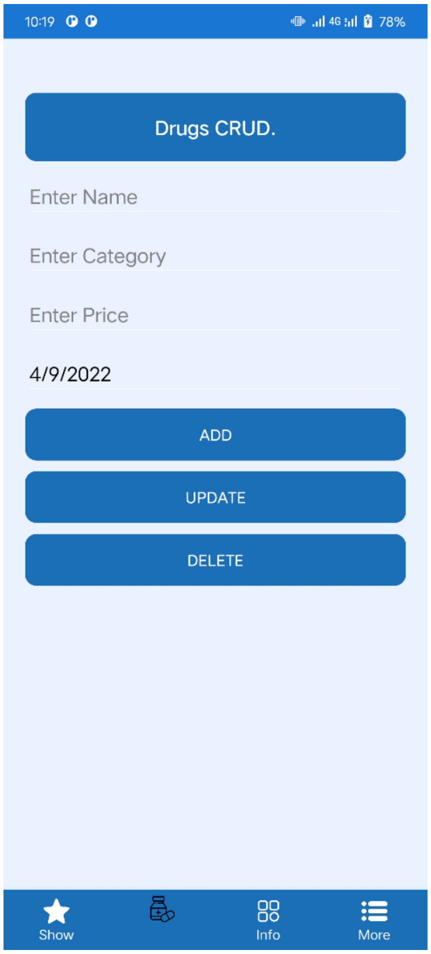


```
0 references
public Drug()
{
    InitializeComponent();
}
1 reference
protected override async void OnAppearing()
{
    base.OnAppearing();
    collectionView.ItemsSource = await App.Database.GETDrugAsync();
}
```

Database.cs



```
6 references
public Task<List<Ddrug>> GETDrugAsync()
{
    var list = _database.Table<Ddrug>().ToListAsync();
    return _database.Table<Ddrug>().ToListAsync();
}
```



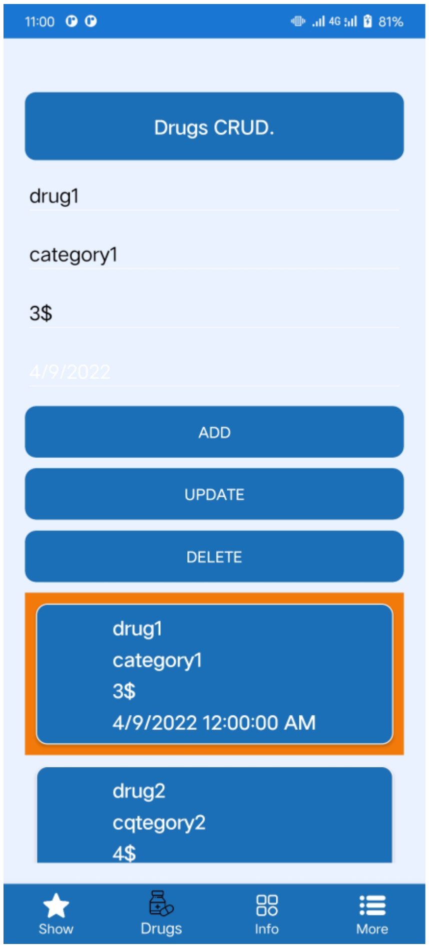
Drugs CRUD.

Enter Name

Enter Category

Enter Price

4/9/2022



Drugs CRUD.

drug1

category1

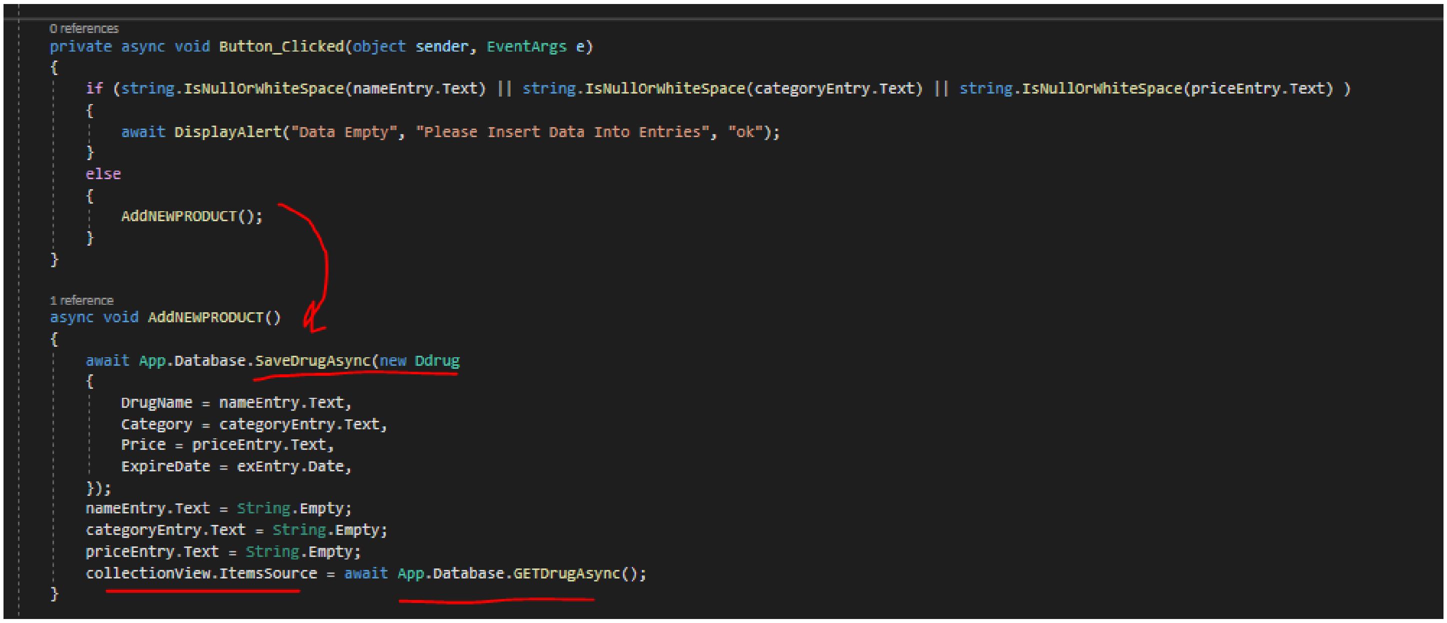
3$

4/9/2022

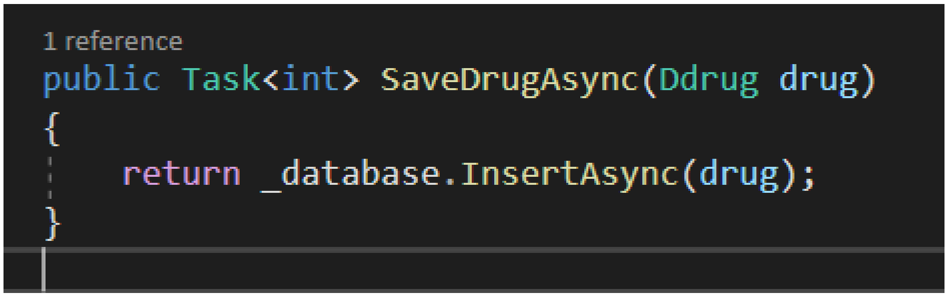
drug1
category1
3$
4/9/2022 12:00:00 AM

drug2
cqtegory2
4$

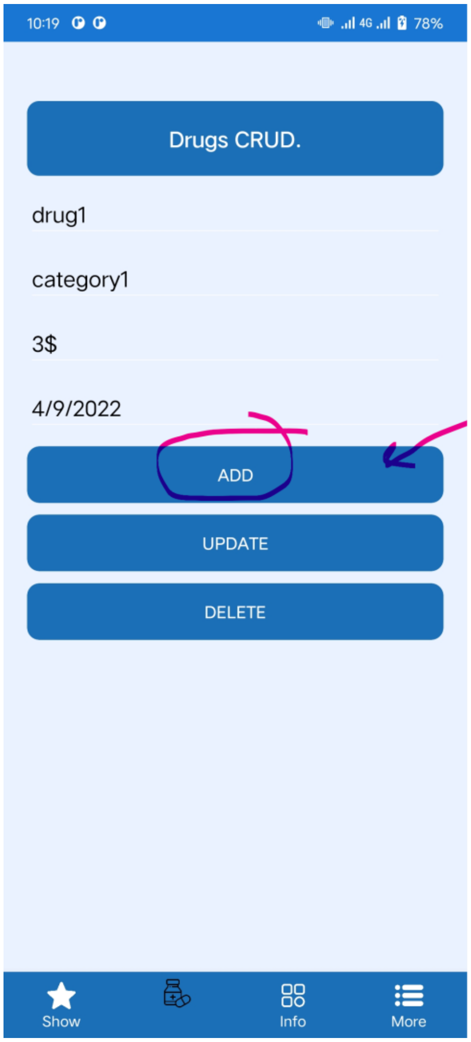
Create Drugs Item



```
0 references
private async void Button_Clicked(object sender, EventArgs e)
{
    if (string.IsNullOrWhiteSpace(nameEntry.Text) || string.IsNullOrWhiteSpace(categoryEntry.Text) || string.IsNullOrWhiteSpace(priceEntry.Text) )
    {
        await DisplayAlert("Data Empty", "Please Insert Data Into Entries", "ok");
    }
    else
    {
        AddNEWPRODUCT();
    }
}

1 reference
async void AddNEWPRODUCT()
{
    await App.Database.SaveDrugAsync(new Ddrug
    {
        DrugName = nameEntry.Text,
        Category = categoryEntry.Text,
        Price = priceEntry.Text,
        ExpireDate = exEntry.Date,
    });
    nameEntry.Text = String.Empty;
    categoryEntry.Text = String.Empty;
    priceEntry.Text = String.Empty;
    collectionView.ItemsSource = await App.Database.GETDrugAsync();
}
```



```
1 reference
public Task<int> SaveDrugAsync(Ddrug drug)
{
    return _database.InsertAsync(drug);
}
```

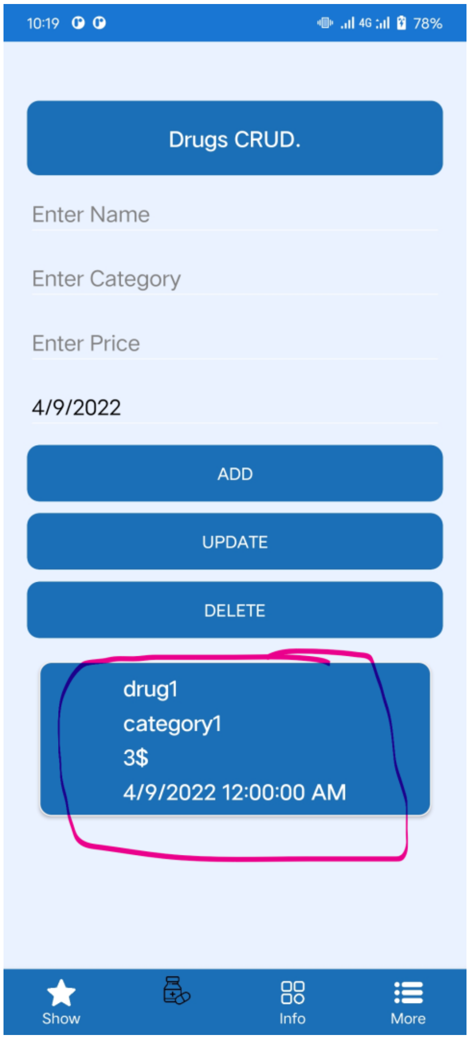


drug1

category1

3$

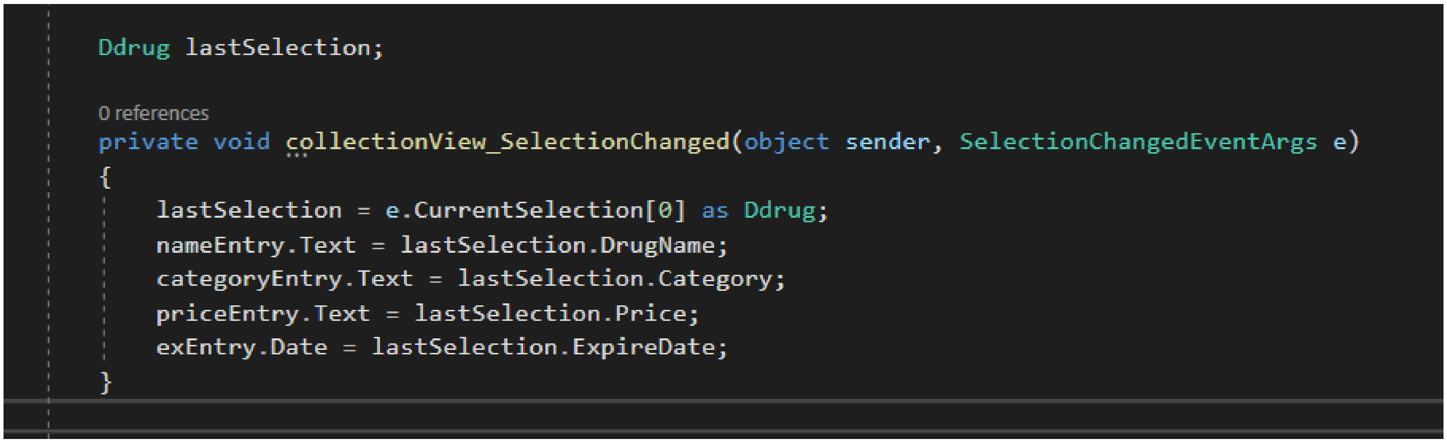
4/9/2022



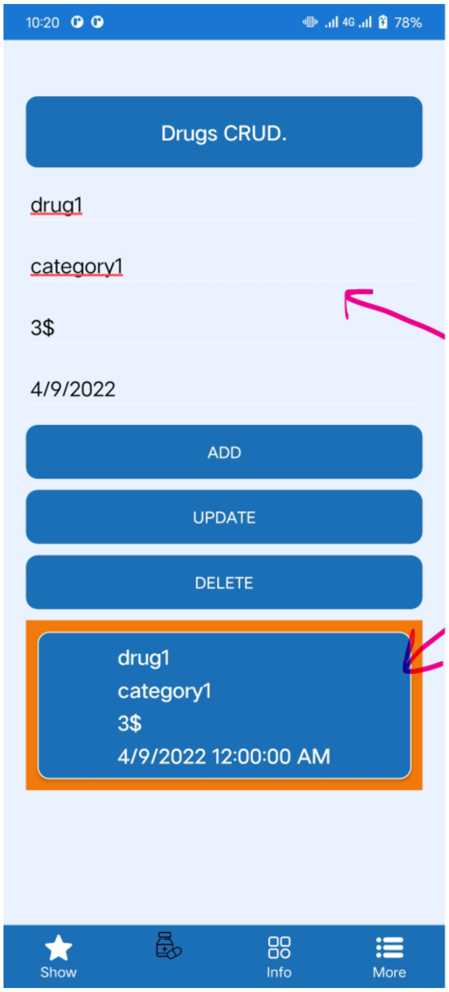
4/9/2022

drug1
category1
3$
4/9/2022 12:00:00 AM

Selection drug item



```
Ddrug lastSelection;

0 references
private void collectionView_SelectionChanged(object sender, SelectionChangedEventArgs e)
{
    lastSelection = e.CurrentSelection[0] as Ddrug;
    nameEntry.Text = lastSelection.DrugName;
    categoryEntry.Text = lastSelection.Category;
    priceEntry.Text = lastSelection.Price;
    exEntry.Date = lastSelection.ExpireDate;
}
```



drug1

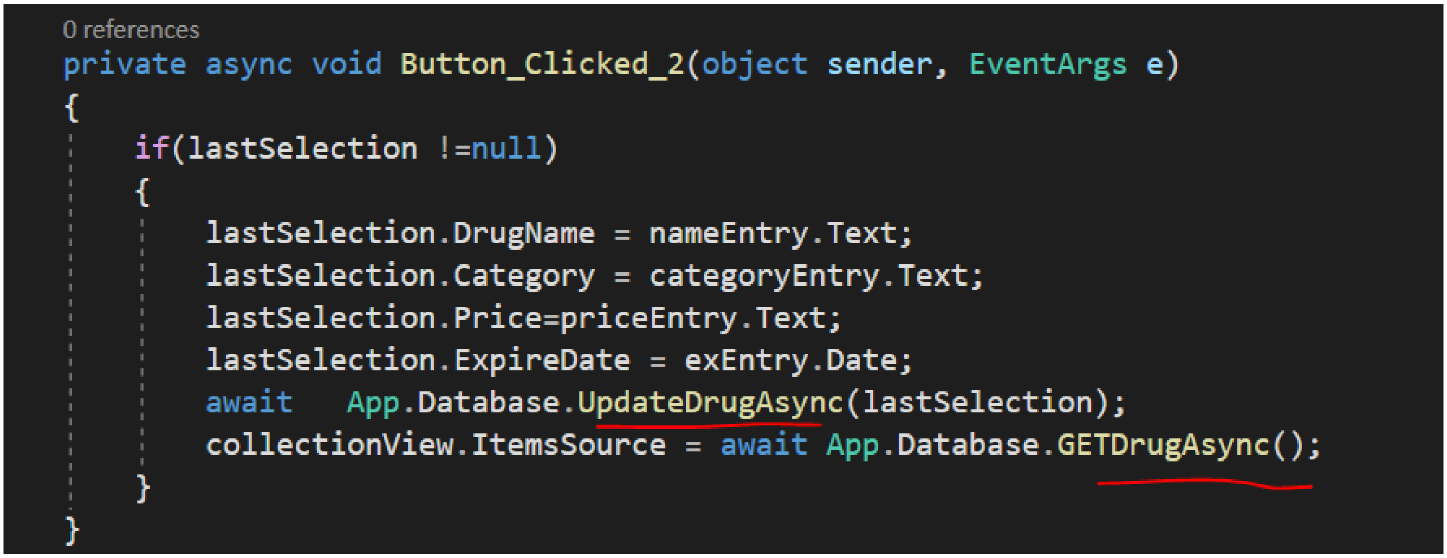
category1

3$

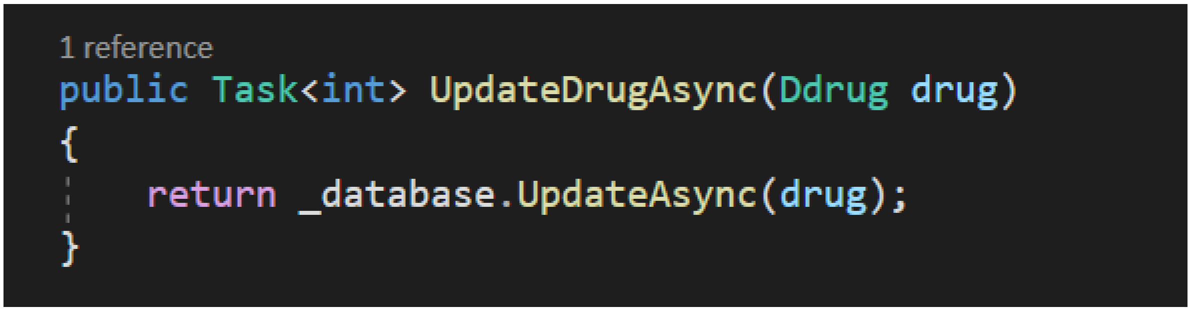
4/9/2022

drug1
category1
3$
4/9/2022 12:00:00 AM

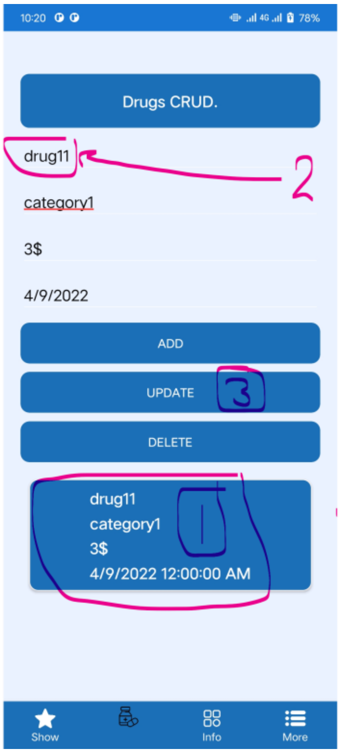
Update drugs



```
0 references
private async void Button_Clicked_2(object sender, EventArgs e)
{
    if(lastSelection !=null)
    {
        lastSelection.DrugName = nameEntry.Text;
        lastSelection.Category = categoryEntry.Text;
        lastSelection.Price=priceEntry.Text;
        lastSelection.ExpireDate = exEntry.Date;
        await   App.Database.UpdateDrugAsync(lastSelection);
        collectionView.ItemsSource = await App.Database.GETDrugAsync();
    }
}
```



```
1 reference
public Task<int> UpdateDrugAsync(Ddrug drug)
{
    return _database.UpdateAsync(drug);
}
```



drug11

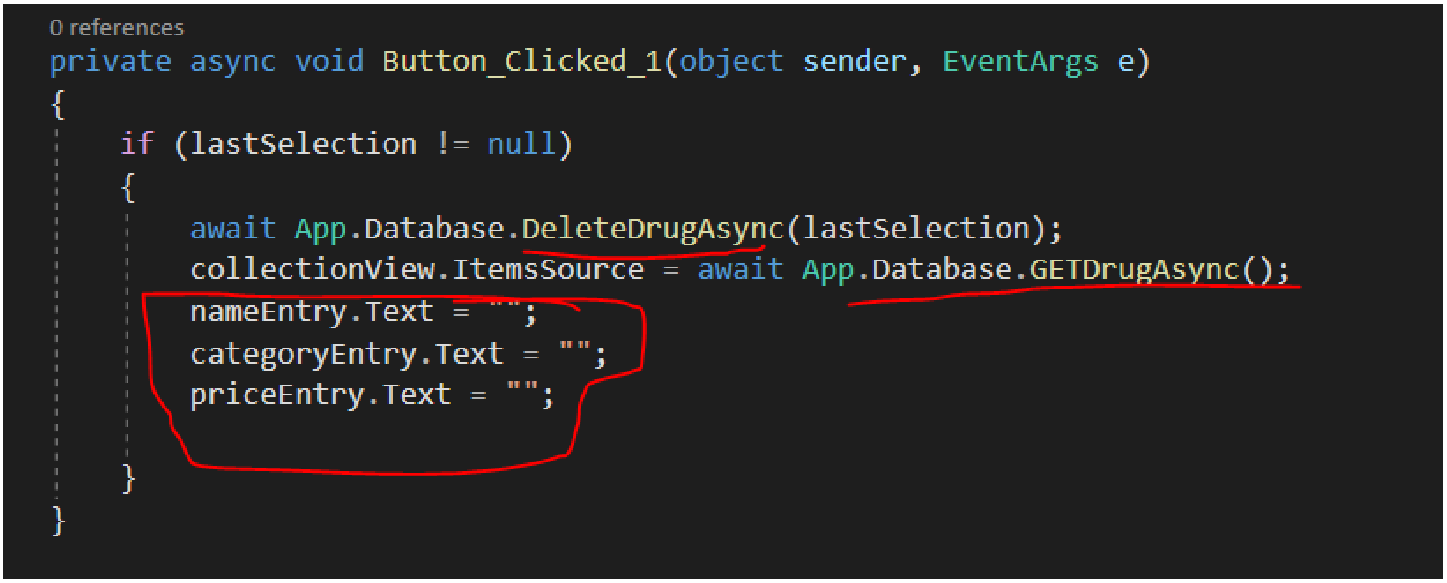
category1

3$

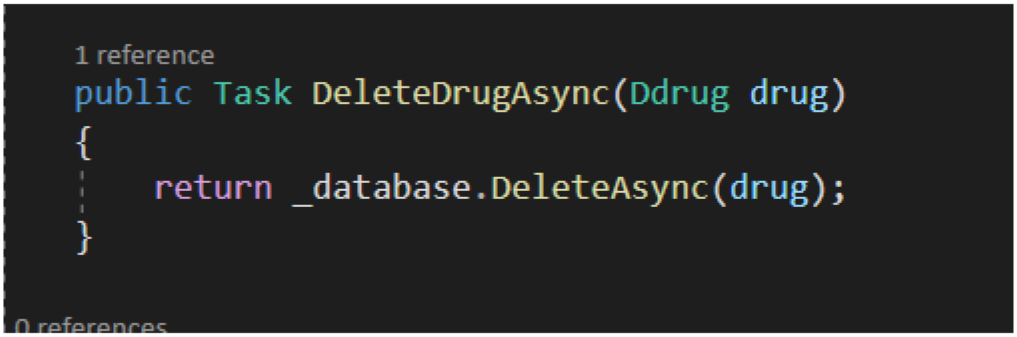
4/9/2022

drug11
category1
3$
4/9/2022 12:00:00 AM

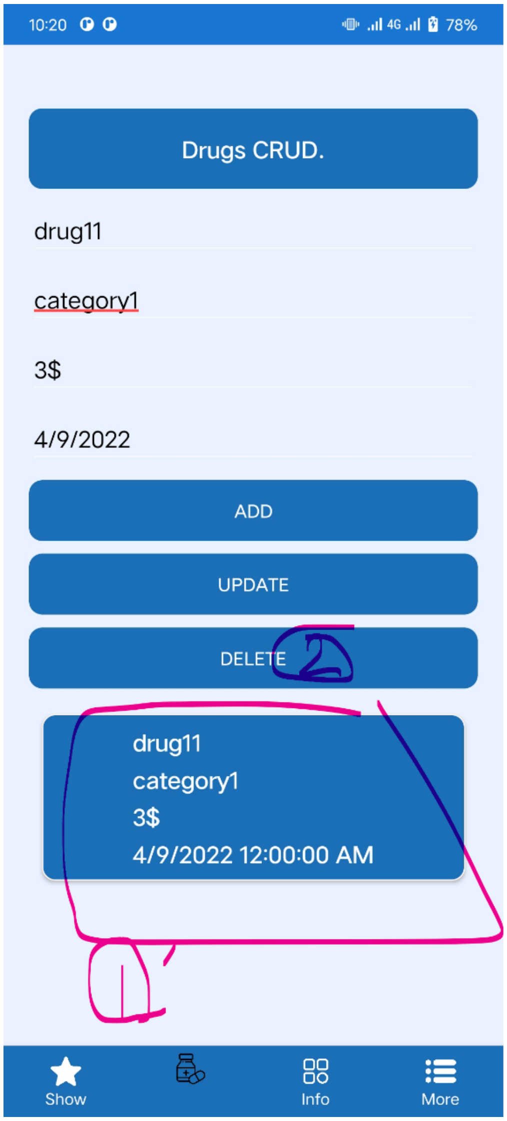
Delete drug



```
0 references
private async void Button_Clicked_1(object sender, EventArgs e)
{
    if (lastSelection != null)
    {
        await App.Database.DeleteDrugAsync(lastSelection);
        collectionView.ItemsSource = await App.Database.GETDrugAsync();
        nameEntry.Text = "";
        categoryEntry.Text = "";
        priceEntry.Text = "";

    }
}
```



```
    1 reference
    public Task DeleteDrugAsync(Ddrug drug)
    {
        return _database.DeleteAsync(drug);
    }

0 references
```



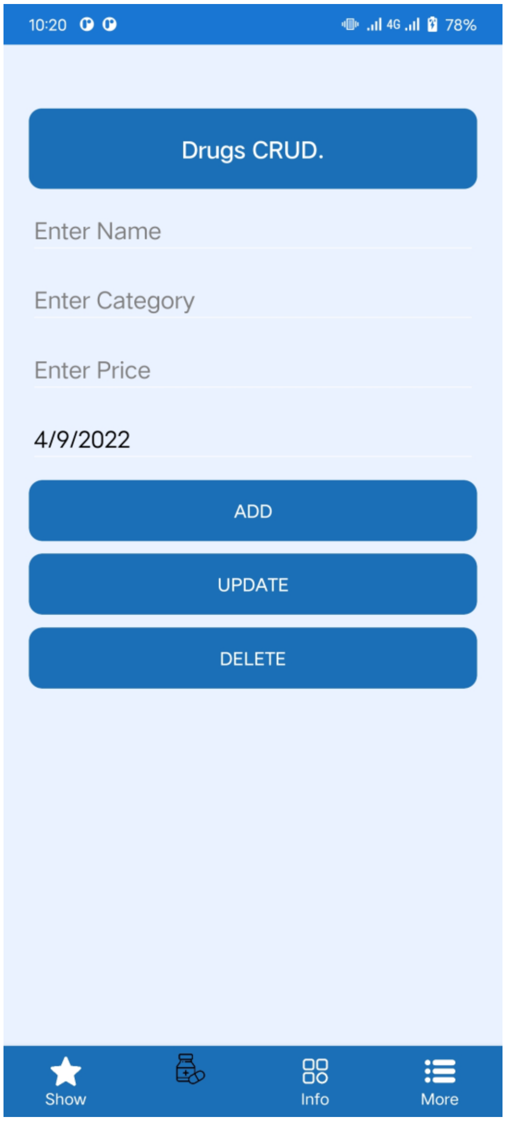
drug11

category1

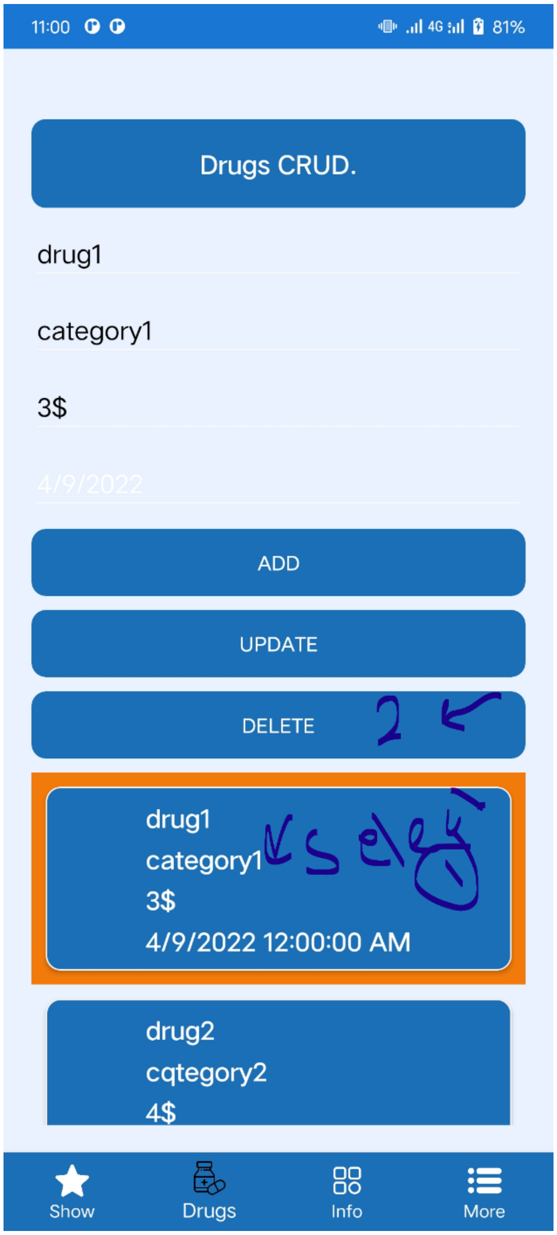
3$

4/9/2022

drug11
category1
3$
4/9/2022 12:00:00 AM



4/9/2022



Drugs CRUD.

drug1

category1

3$

4/9/2022

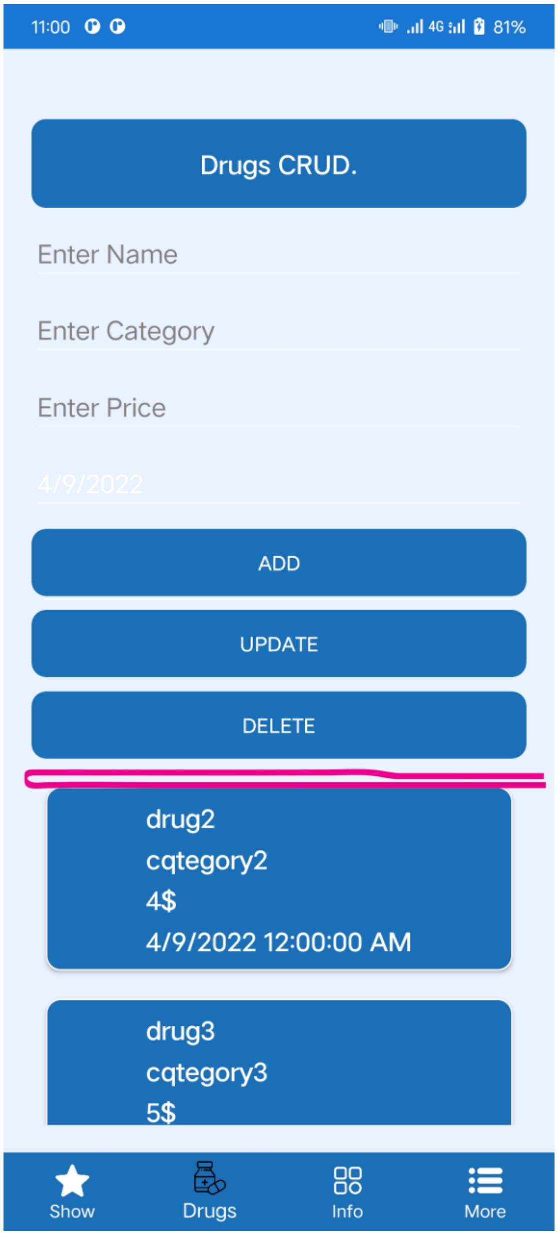
ADD

UPDATE

DELETE

drug1
category1
3$
4/9/2022 12:00:00 AM

drug2
cqtegory2
4$



Drugs CRUD.

Enter Name

Enter Category

Enter Price

4/9/2022

ADD

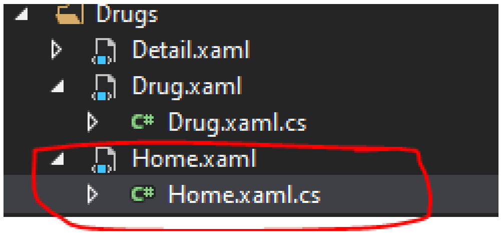
UPDATE

DELETE

drug2
cqtegory2
4$
4/9/2022 12:00:00 AM

drug3
cqtegory3
5$

Home Page



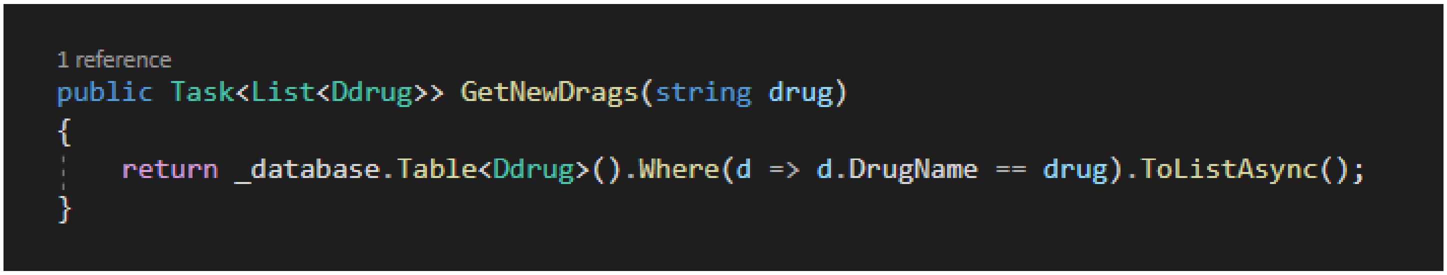
Drugs
Detail.xaml
Drug.xaml
Drug.xaml.cs
Home.xaml
Home.xaml.cs

This file to show all drugs and you can easily search to a specific drug and you can also see more details about the drug from **Detail**.cs page!

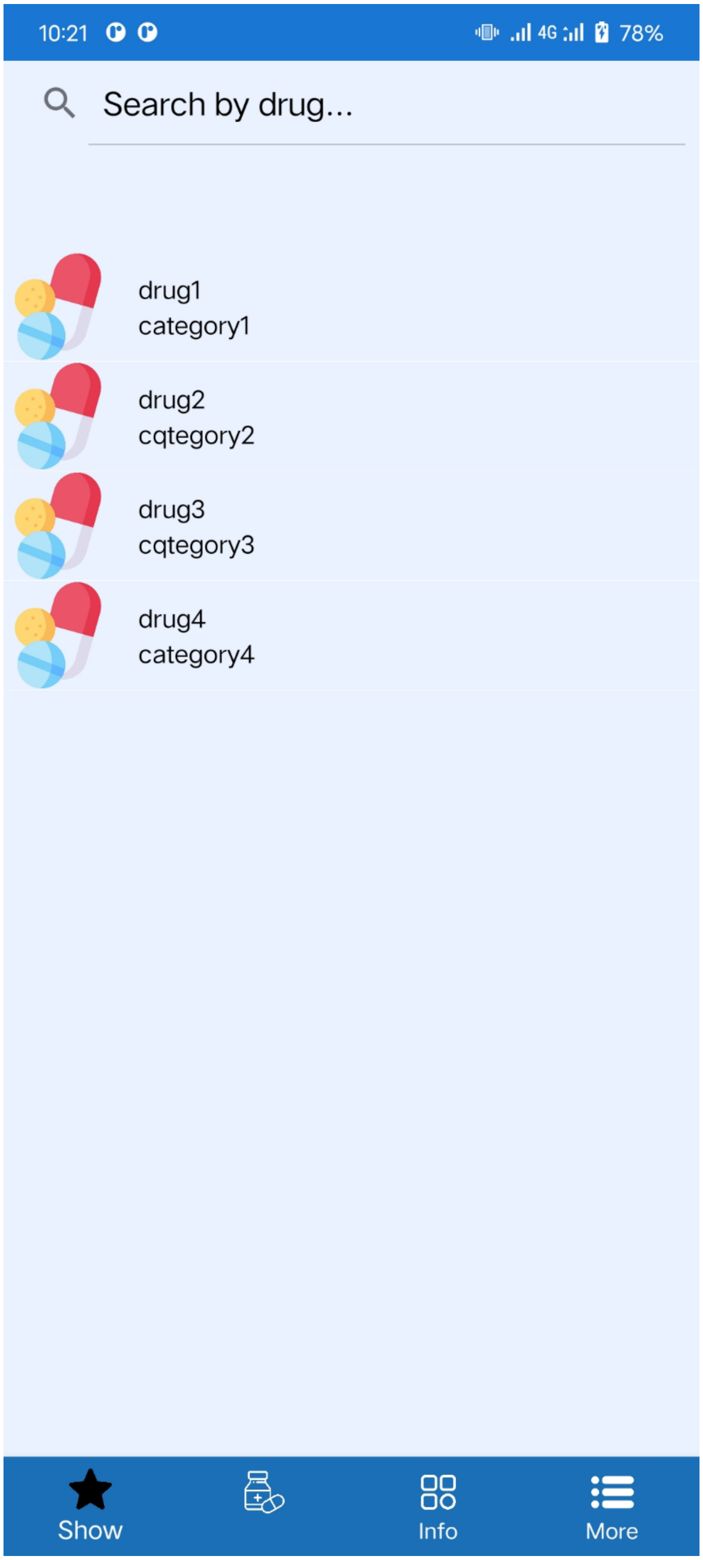
```
        private BindableProperty IsSearchingProperty =
         BindableProperty.Create("IsSearching", typeof(bool), typeof(Home), false);
         public bool IsSearching
         {
             get { return (bool)GetValue(IsSearchingProperty); }
             set { SetValue(IsSearchingProperty, value); }
         }
         public Home()
         {
             BindingContext = this;
             InitializeComponent();
         }

         protected override async void OnAppearing()
         {
             base.OnAppearing();
             listView.ItemsSource = await App.Database.GETDrugAsync();
         }

         private async void listView_ItemSelected(object sender, SelectedItemChangedEventArgs e)
         {

             if (e.SelectedItem == null)
                 return;
             var drug = e.SelectedItem as Ddrug;
             await Navigation.PushAsync(new Detail(drug));
             listView.SelectedItem = null;
         }

         private async void SearchBar_TextChanged(object sender, TextChangedEventArgs e)
         {

             if (e.NewTextValue == null || e.NewTextValue.Length <1)
                 listView.ItemsSource = await App.Database.GETDrugAsync();
```

```
            await FindDrugs(actor: e.NewTextValue);
        }

        async Task FindDrugs(string actor)
        {

            try
            {
                IsSearching = true;
                var drugs = await App.Database.GetNewDrags(actor);
                listView.ItemsSource = drugs;
                listView.IsVisible = drugs.Any();
                notFound.IsVisible = !listView.IsVisible;
            }
            catch (Exception)
            {
                await DisplayAlert("Error", "Could not retrieve the list of Drgs.", "OK");
            }
            finally
            {
                IsSearching = false;

            }
        }
```

Database.cs



```
1 reference
public Task<List<Ddrug>> GetNewDrags(string drug)
{
    return _database.Table<Ddrug>().Where(d => d.DrugName == drug).ToListAsync();
}
```



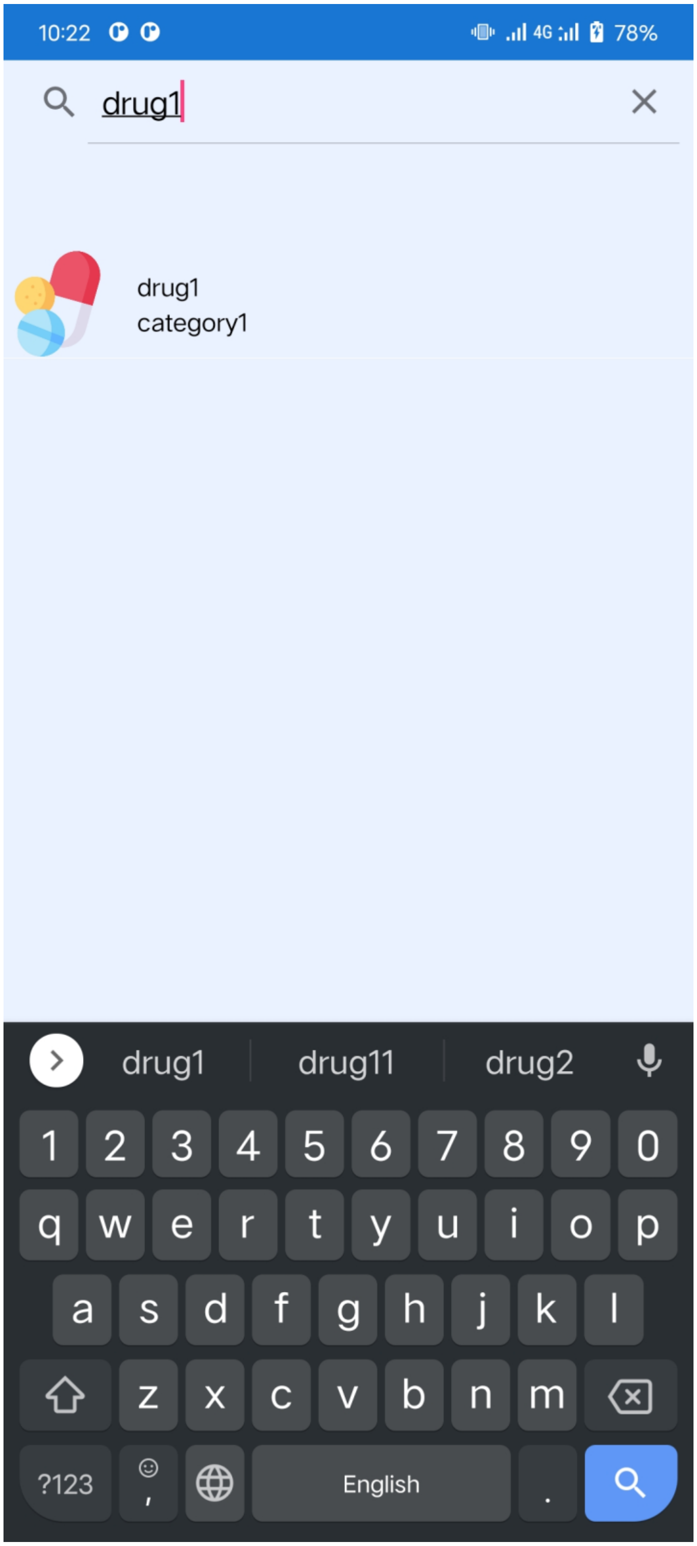
Search by drug...

drug1
category1

drug2
cqtegory2

drug3
cqtegory3

drug4
category4

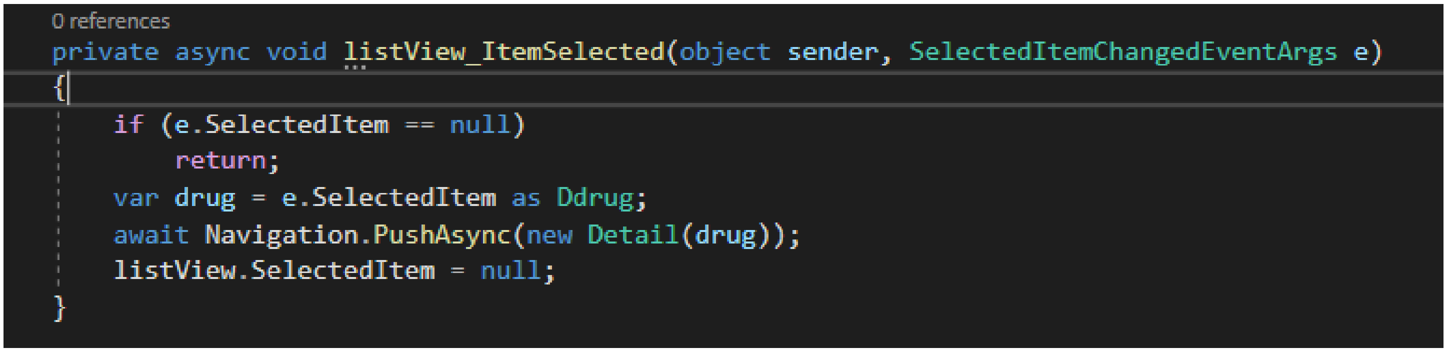


drug1

drug1
category1

Detail page

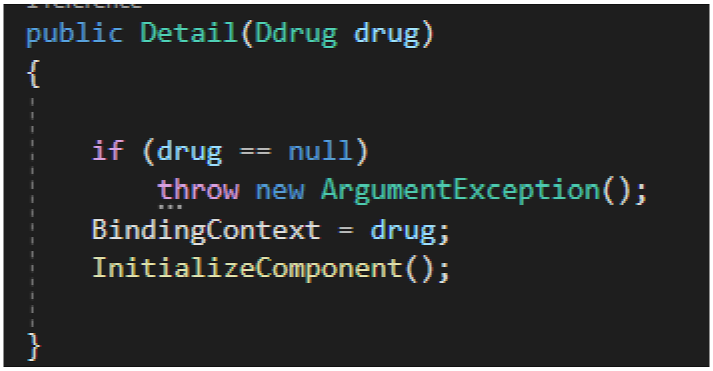
Drug.cs



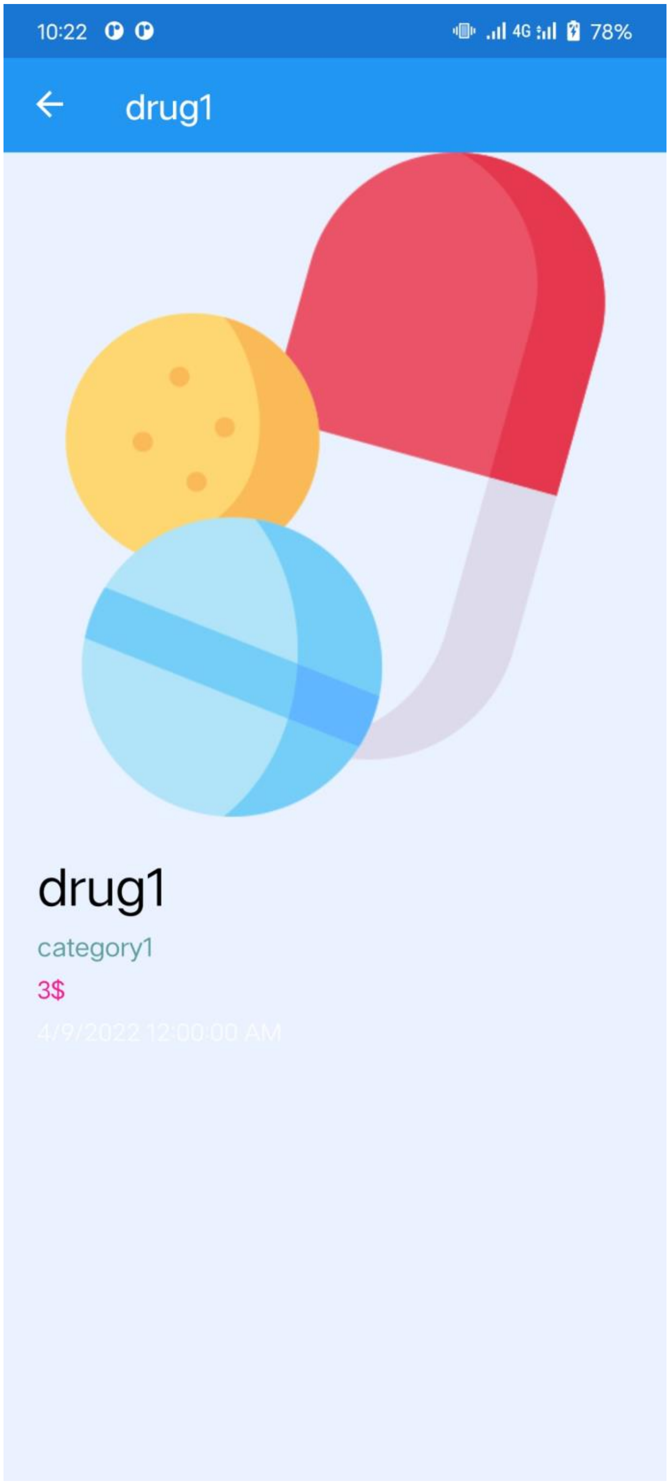
```
0 references
private async void listView_ItemSelected(object sender, SelectedItemChangedEventArgs e)
{
    if (e.SelectedItem == null)
        return;
    var drug = e.SelectedItem as Ddrug;
    await Navigation.PushAsync(new Detail(drug));
    listView.SelectedItem = null;
}
```

This method let you select an drugs to show you more detail

And send current selection object data to Detail.cs



```
public Detail(Ddrug drug)
{

    if (drug == null)
        throw new ArgumentException();
    BindingContext = drug;
    InitializeComponent();

}
```



drug1

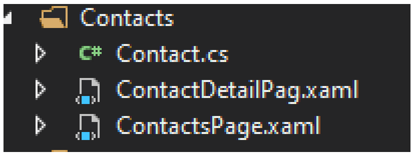
drug1

category1

3$

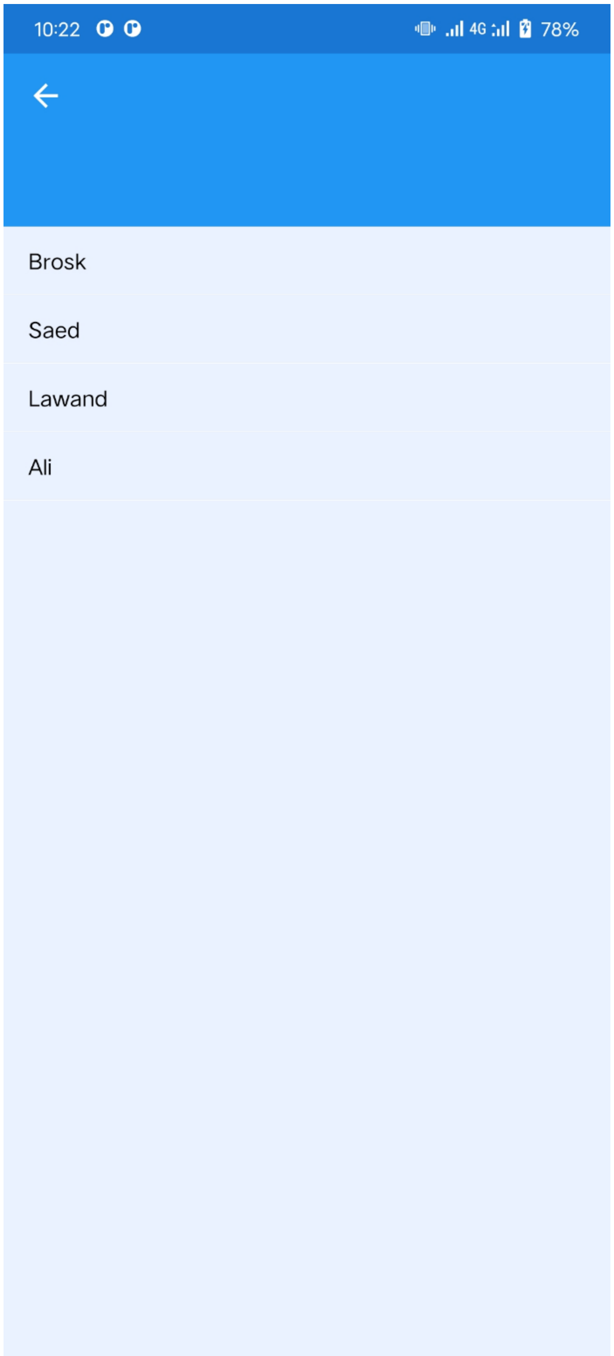
4/9/2022 12:00:00 AM

Contact Foldar



```
Contacts
    Contact.cs
    ContactDetailPag.xaml
    ContactsPage.xaml
```

Same way to contact page and show contact detail



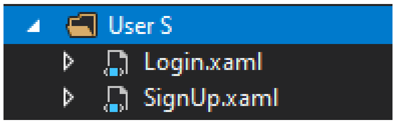
Brosk

Saed

Lawand

Ali

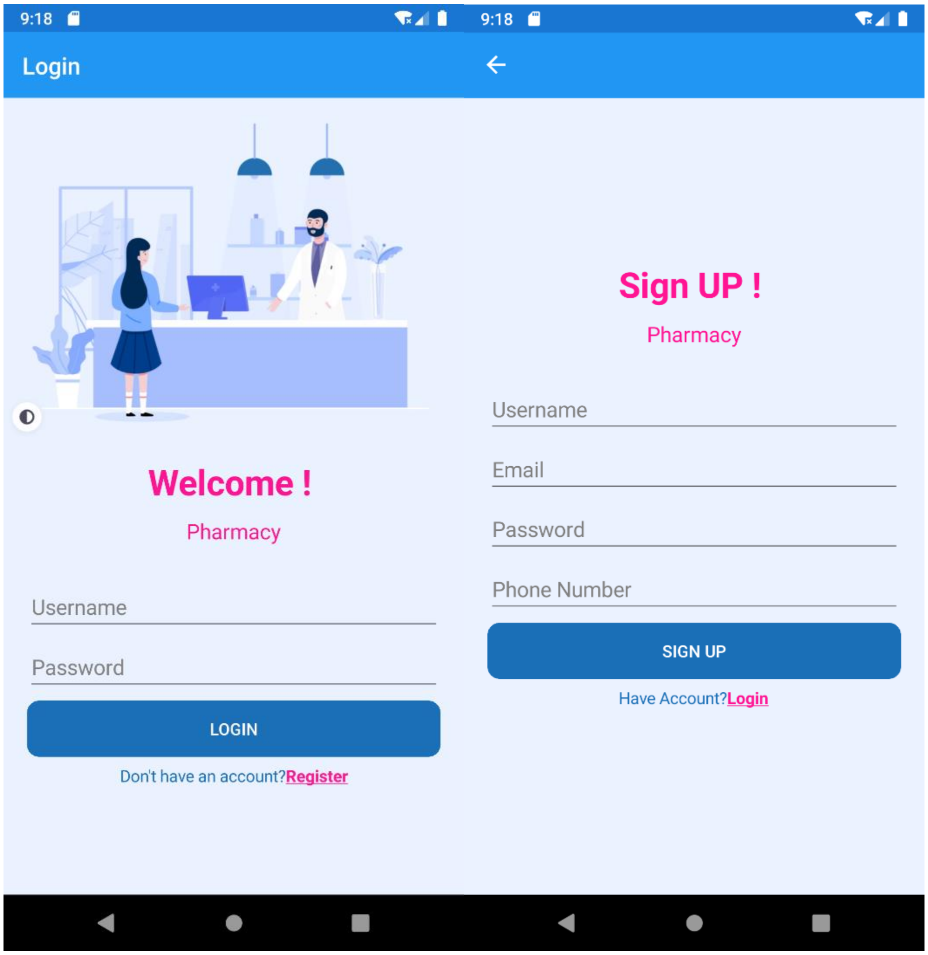
System Login



User S

Login.xaml

SignUp.xaml



9:18

Login

Welcome !

Pharmacy

Username

Password

LOGIN

Don't have an account?Register

9:18

Sign UP !

Pharmacy

Username

Email

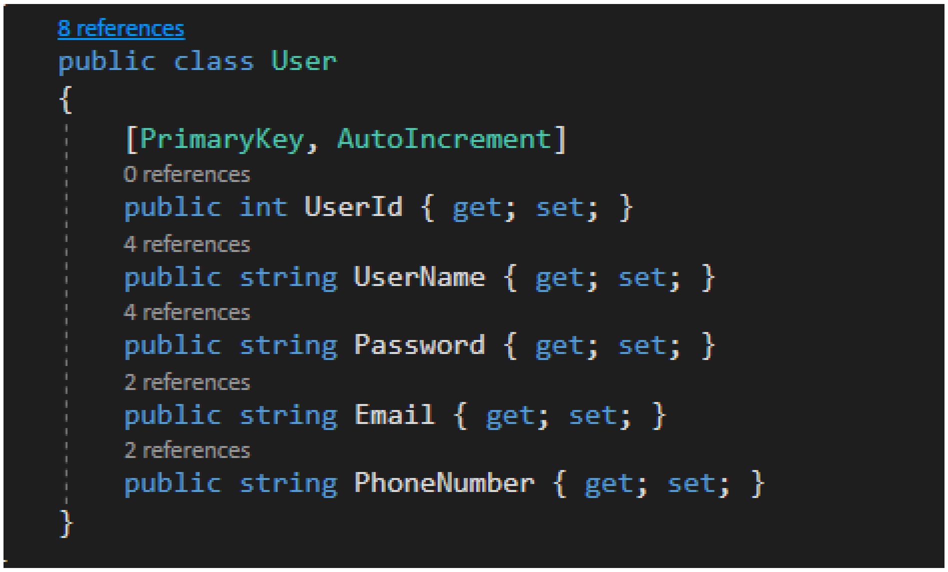
Password

Phone Number

SIGN UP

Have Account?Login

User.cs

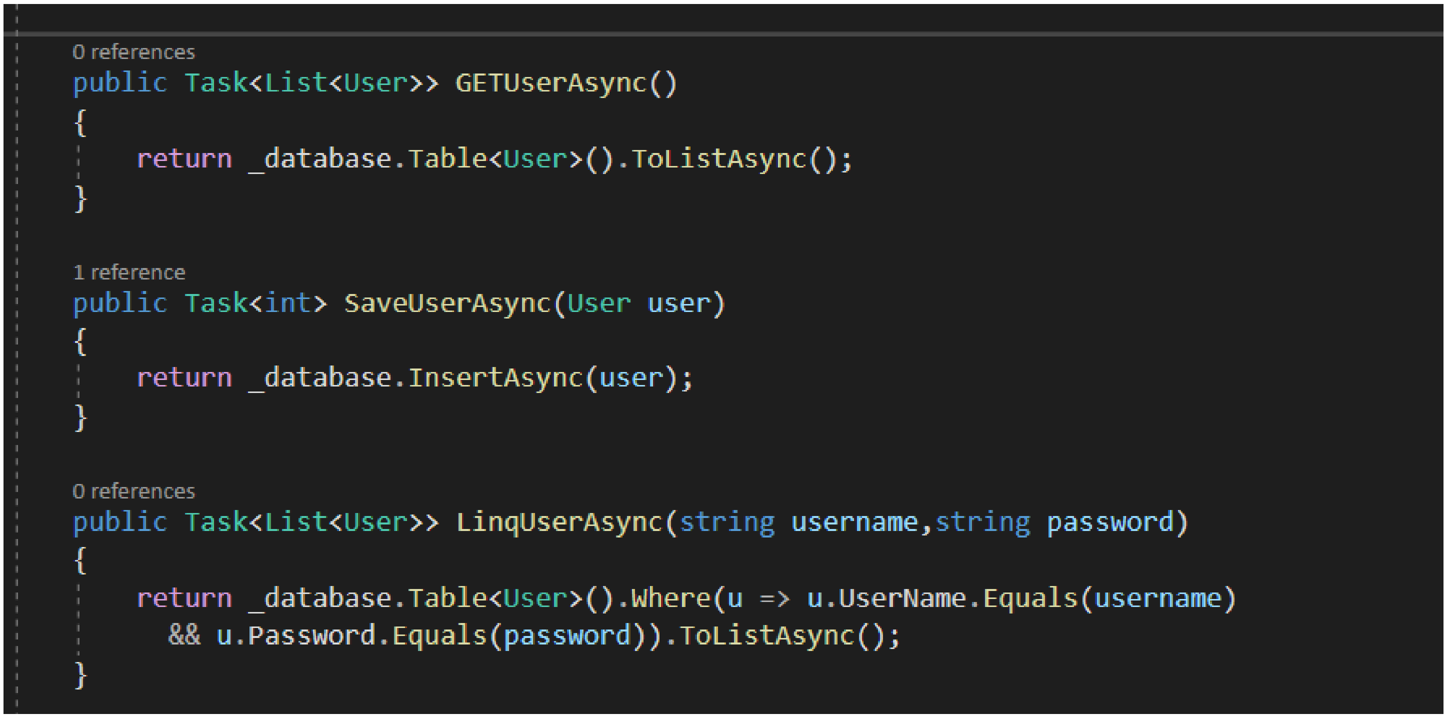


```
8 references
public class User
{
    [PrimaryKey, AutoIncrement]
    0 references
    public int UserId { get; set; }
    4 references
    public string UserName { get; set; }
    4 references
    public string Password { get; set; }
    2 references
    public string Email { get; set; }
    2 references
    public string PhoneNumber { get; set; }
}
```

User Tabel

We created this table to login system and to shows users profile
And secure a specific data from database

Database.cs



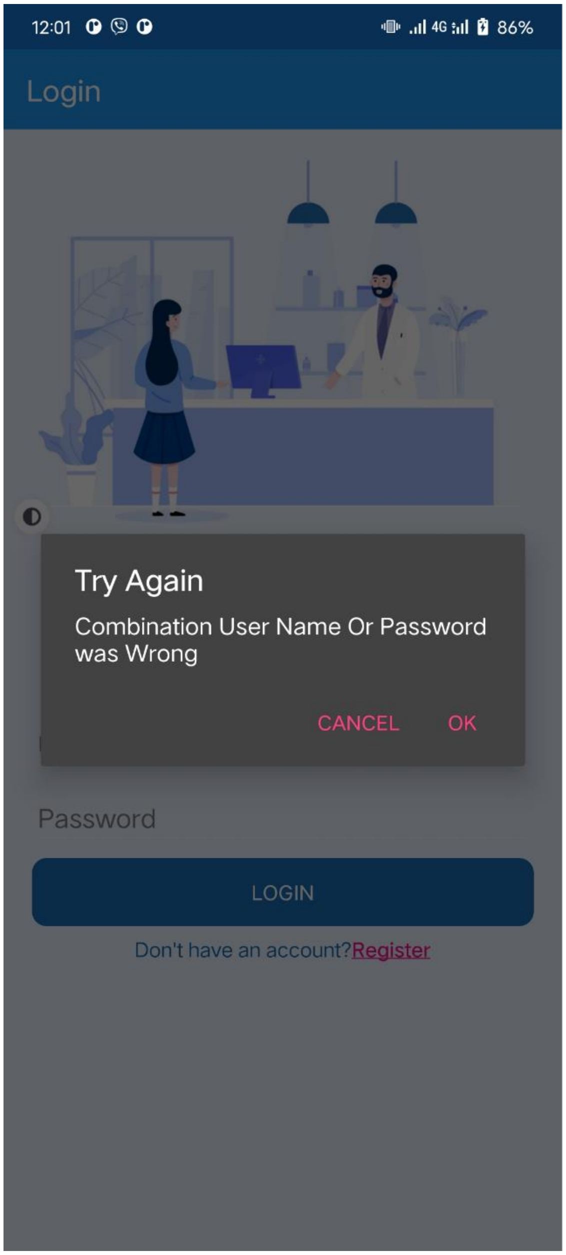
```
0 references
public Task<List<User>> GETUserAsync()
{
    return _database.Table<User>().ToListAsync();
}

1 reference
public Task<int> SaveUserAsync(User user)
{
    return _database.InsertAsync(user);
}

0 references
public Task<List<User>> LinqUserAsync(string username,string password)
{
    return _database.Table<User>().Where(u => u.UserName.Equals(username)
      && u.Password.Equals(password)).ToListAsync();
}
```

We can use the same way to cretae new user and get a uniqe user from database And check is user exist or not and we can easely send data to application profile and user can easly change profile

Another way

Login

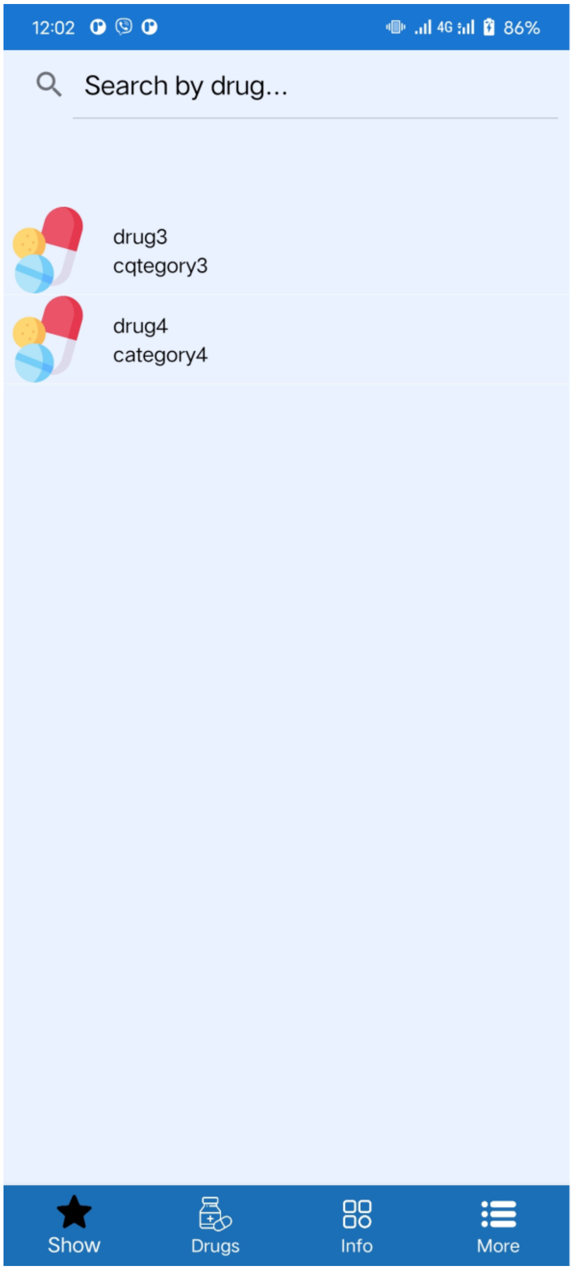
```
  public Login()
        {
            InitializeComponent();
        }


        async private void btnLogin_Clicked(object sender, EventArgs e)
        {



              var dbpath =
Path.Combine(Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData), "User.db");
              var db = new SQLiteConnection(dbpath);
              var myquery = db.Table<User>().Where(u => u.UserName.Equals(username.Text)
              && u.Password.Equals(password.Text)).FirstOrDefault();

              if (myquery != null)
              {
                await Navigation.PushAsync(new MainPage());
            }

              else
              {
                  Device.BeginInvokeOnMainThread(async () =>
                  {
                      var result = await this.DisplayAlert("Try Again", "Combination User Name Or
Password was Wrong", "Ok", "Cancel");
                      if (result) await Navigation.PushAsync(new Login());
                      else
                      {
                          await Navigation.PushAsync(new Login());
                      }

                  });
                  username.Text = String.Empty;
                  password.Text = String.Empty;

            }
        }

        private void TapGestureRecognizer_Tapped(object sender, EventArgs e)
        {
            Navigation.PushAsync(new SignUp());
}
```



Try Again

Combination User Name Or Password was Wrong

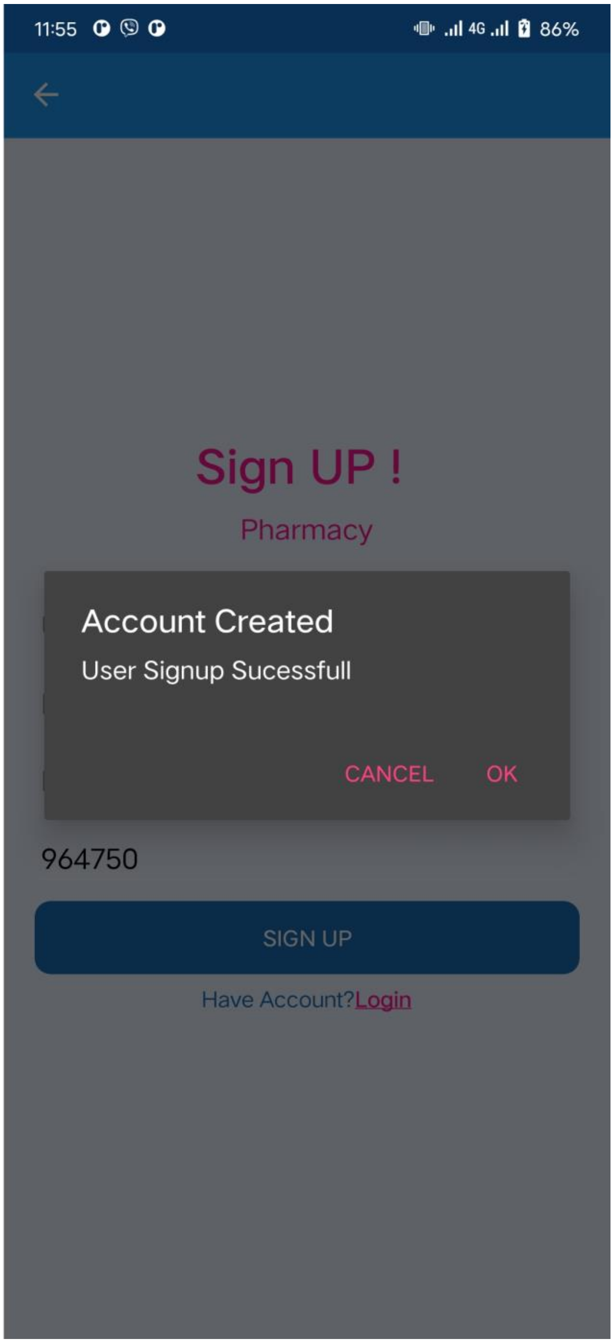
Don't have an account?Register



drug3	cqtegory3
drug4	category4

Sign up

```
   public SignUp()
        {
            InitializeComponent();
        }

        private async void Button_Clicked(object sender, EventArgs e)
        {

            if (string.IsNullOrWhiteSpace(EntryUsername.Text) ||
string.IsNullOrWhiteSpace(EntryEmail.Text) || string.IsNullOrWhiteSpace(EntryPassword.Text) ||
string.IsNullOrWhiteSpace(EntryPhonenumber.Text))
            {
                await DisplayAlert("Empty Filds", "Please Insert Info Into Entries ", "ok");
            }
            else
            {
                //   AddNewUser();
                var dbpath =
Path.Combine(Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData), "User.db");
                var db = new SQLiteConnection(dbpath);
                db.CreateTable<User>();

                var user = new User()
                {
                    UserName = EntryUsername.Text,
                    Password = EntryPassword.Text,
                    Email = EntryEmail.Text,
                    PhoneNumber = EntryPhonenumber.Text
                };

                db.Insert(user);

                Device.BeginInvokeOnMainThread(async () =>
                {
                    var result = await this.DisplayAlert("Account Created", "User Signup Sucessfull",
"Ok", "Cancel");
                    await Navigation.PushAsync(new MainPage());
                });

                  EntryUsername.Text = String.Empty;
                  EntryEmail.Text = String.Empty;
                  EntryPassword.Text = String.Empty;
            }

        }
       async void AddNewUser()
        {
            await App.Database.SaveUserAsync(new User
            {
                UserName = EntryUsername.Text,
                Email = EntryEmail.Text,
                Password = EntryPassword.Text,
                PhoneNumber = EntryPhonenumber.Text,
            });
            Device.BeginInvokeOnMainThread(async () =>
            {
                var result = await this.DisplayAlert("Account Created", "User Signup Sucessfull", "Ok",
"Cancel");
                await Navigation.PushAsync(new MainPage());
            });
        }
        private void TapGestureRecognizer_Tapped(object sender, EventArgs e)
        {
            Navigation.PushAsync(new Login());
        }
```

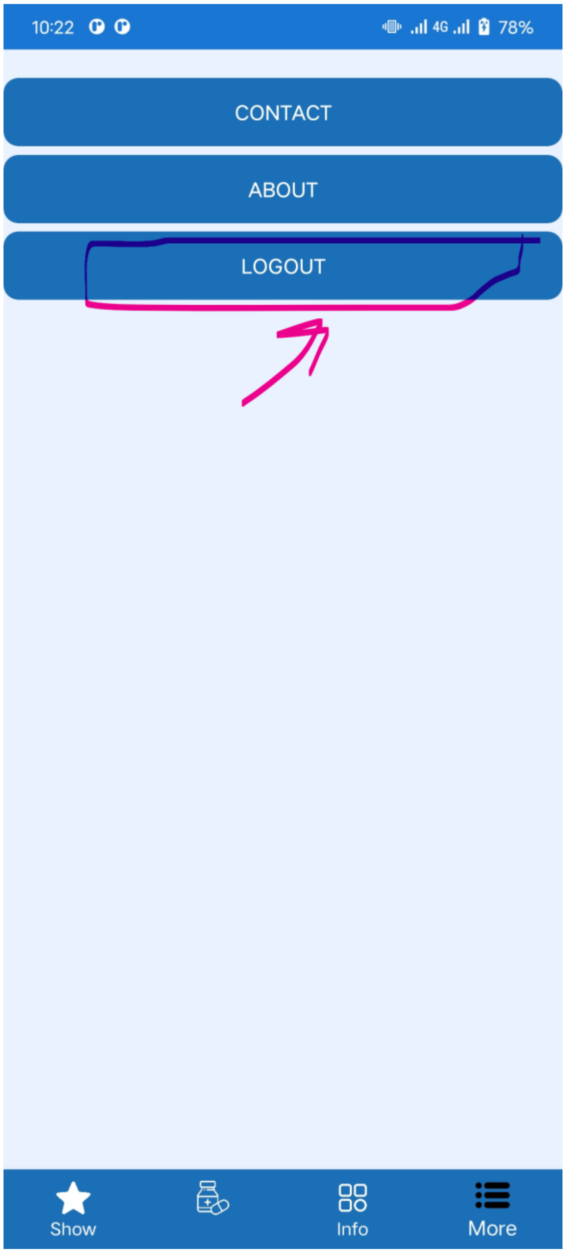


Account Created

User Signup Sucessfull

If users username and password is correct will open the main page other wise it will show error.

Logout



```
await Navigation.PopToRootAsync();
```

Chapter 4

Conclusion and Feature Work

Admin Features Previous features were targeted at drug buyers, however, there is another group of users either, namely the pharmacy staff. Your employees should be able to manage application content, update data, and perform other similar functions.

References

1. "Xamarin.forms an open-source UI framework: .NET," *Microsoft*. [Online]. Available: https://dotnet.microsoft.com/en-us/apps/xamarin/xamarin-forms. [Accessed: 09-Apr-2022].

2. "Introduction To Xamarin For Beginners," *www.c-sharpcorner.com*. https://www.c-sharpcorner.com/article/introduction-to-xamarin-for-beginners/ (accessed Apr. 09, 2022).