Лабораторная работа №14

Работа с SQLite

Цель работы: получение навыков разработки приложения для работы с SQLite.

Теоретическая часть

Для хранения данных мобильного приложения можно использовать не только файлы, но и локальную базу данных. Наиболее популярной СУБД для локальных приложений является SQLite.

Шаг 1

Создайте пустой проект с именем `SupplyMobileApp`. Тип проекта – `Mobile App (Xamarin.Forms)`.

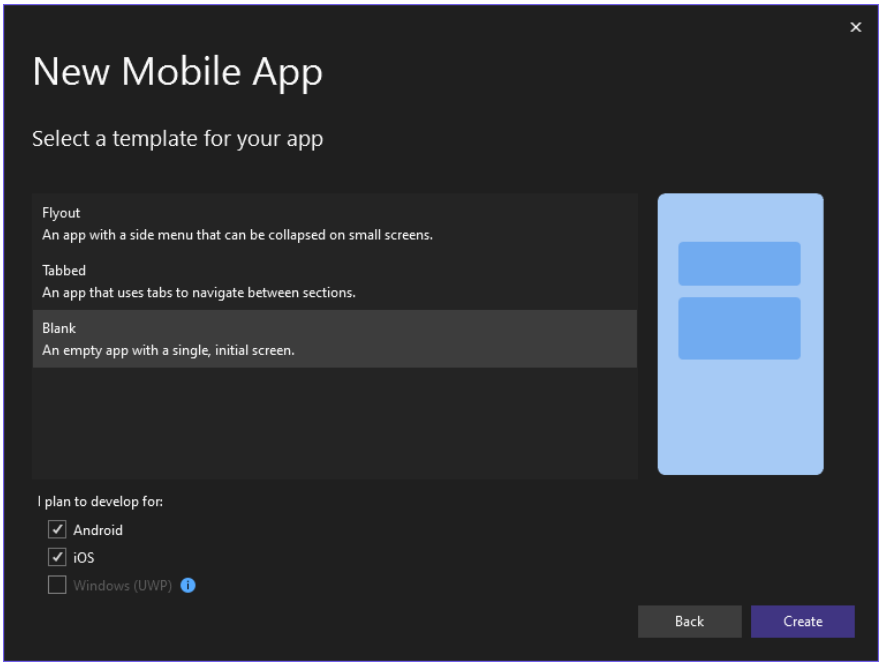

Рисунок 1 – Создание проекта

Шаг 2

Далее нужно добавить в проект библиотеку для работы с SQLite через диспетчер пакетов NuGet. Таких библиотек существует большое количество, для примера можно взять простую библиотеку `sqlite-net-pcl`.

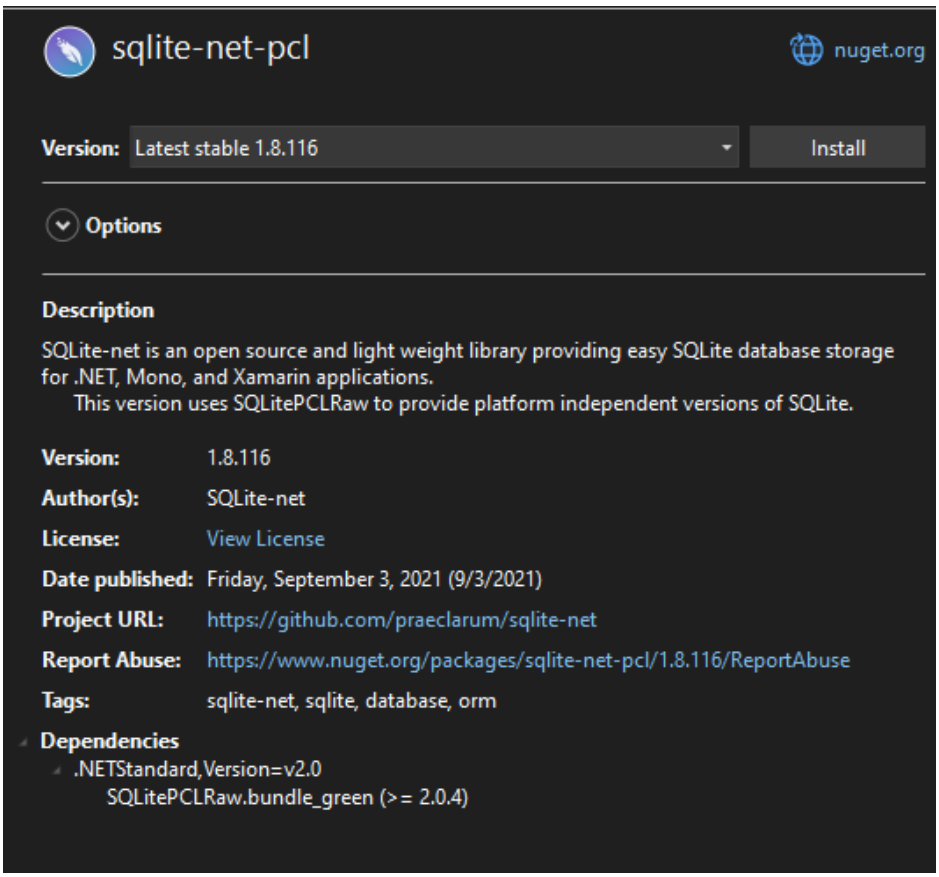

Рисунок 2 – Добавление библиотеки

Шаг 3

В данном примере рассмотрим работу с таблицей Item из БД Supply.

Добавьте в проект класс `Item`.

```
using SQLite;

namespace SupplyMobileApp
{
    [Table("Item")]
    public class Item
    {
        [PrimaryKey, AutoIncrement]
        public int Id { get; set; }

        [MaxLength(50)]
        public string Name { get; set; }

        [MaxLength(50)]
        public string Manufacturer { get; set; }

        public decimal Price { get; set; }
    }
}
```

Шаг 4

Теперь нужно добавить класс `ItemDataService`, в котором будет описана логика работы с БД.

```
using SQLite;
using System.Collections.Generic;

namespace SupplyMobileApp
{
    public class ItemDataService
    {
        SQLiteConnection _database;

        public ItemDataService(string databasePath)
        {
            _database = new SQLiteConnection(databasePath);
            _database.CreateTable<Item>();
        }

        public IEnumerable<Item> GetItems()
        {
            return _database.Table<Item>().ToList();
        }

        public Item GetItem(int id)
        {
            return _database.Get<Item>(id);
        }

        public int DeleteItem(int id)
        {
            return _database.Delete<Item>(id);
        }

        public int SaveItem(Item item)
        {
            if (item.Id != 0)
            {
                _database.Update(item);
                return item.Id;
            }
            else
            {
                return _database.Insert(item);
            }
        }
    }
}
```

В конструкторе класса происходит создание подключения и базы данных (если она отсутствует). В качестве параметра извне будет передаваться путь к базе данных. Также описаны методы для основных операций с данными в таблице.

Шаг 5

Теперь добавим код в файл `App.xaml.cs`.

```
using System;
using System.IO;
using Xamarin.Forms;

namespace SupplyMobileApp
{
    public partial class App : Application
    {
        public const string DATABASE_NAME = "supply.db";
        private static ItemDataService _database;

        public static ItemDataService Database
        {
            get
            {
                if (_database == null)
                {
                    _database = new ItemDataService(
                        Path.Combine(

Environment.GetFolderPath(Environment.SpecialFolder.LocalApplicationData),
DATABASE_NAME));
                }
                return _database;
            }
        }

        public App()
        {
            InitializeComponent();

            MainPage = new NavigationPage(new MainPage());
        }

        protected override void OnStart()
        {
        }

        protected override void OnSleep()
        {
        }

        protected override void OnResume()
        {
        }
    }
}
```

Шаг 6

В данном приложении будет две страницы – страница со списком товаров и страница для работы с отдельным товаром. Поэтому добавим новый элемент `ContentPage` с именем `ItemPage`.

```
<?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="SupplyMobileApp.ItemPage" Title="Информация о товаре">
    <StackLayout>
        <Label Text="Наименование" />
        <Entry Text="{Binding Name}" />
        <Label Text="Производитель" />
        <Entry Text="{Binding Manufacturer}" />
        <Label Text="Цена" />
        <Entry Text="{Binding Price}" />
        <StackLayout Orientation="Horizontal">
            <Button Text="Сохранить" Clicked="SaveItem" />
            <Button Text="Удалить" Clicked="DeleteItem" />
            <Button Text="Отмена" Clicked="Cancel" />
        </StackLayout>
    </StackLayout>
</ContentPage>
```

Для этой страницы добавим обработчики для нажатия кнопок.

```
using System;
using Xamarin.Forms;

namespace SupplyMobileApp
{
    public partial class ItemPage : ContentPage
    {
        public ItemPage()
        {
            InitializeComponent();
        }

        private void SaveItem(object sender, EventArgs e)
        {
            var item = (Item)BindingContext;
            if (!String.IsNullOrEmpty(item.Name))
            {
                App.Database.SaveItem(item);
            }
            Navigation.PopAsync();
        }

        private void DeleteItem(object sender, EventArgs e)
        {
            var item = (Item)BindingContext;
            App.Database.DeleteItem(item.Id);
            Navigation.PopAsync();
        }

        private void Cancel(object sender, EventArgs e)
        {
            Navigation.PopAsync();
        }
    }
}
```

Шаг 7

Теперь изменим содержимое страницы `MainPage`.

```
<?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="SupplyMobileApp.MainPage" Title="Товары">
    <StackLayout>
        <ListView x:Name="itemList" ItemsSource="{Binding}"
ItemSelected="OnItemSelected">
            <ListView.ItemTemplate>
                <DataTemplate>
                    <ViewCell>
                        <ViewCell.View>
                            <StackLayout Orientation="Horizontal">
                                <Label Text="{Binding Name}" FontSize="Medium" />
                            </StackLayout>
                        </ViewCell.View>
                    </ViewCell>
                </DataTemplate>
            </ListView.ItemTemplate>
        </ListView>
        <Button Text="Добавить" Clicked="CreateItem" />
    </StackLayout>
</ContentPage>
```

И добавим обработчики в файл `MainPage.xaml.cs`.

```
using System;
using Xamarin.Forms;

namespace SupplyMobileApp
{
    public partial class MainPage : ContentPage
    {
        public MainPage()
        {
            InitializeComponent();
        }

        protected override void OnAppearing()
        {
            itemList.ItemsSource = App.Database.GetItems();
            base.OnAppearing();
        }

        private async void OnItemSelected(object sender,
SelectedItemChangedEventArgs e)
        {
            Item selectedItem = (Item)e.SelectedItem;
            ItemPage itemPage = new ItemPage();
            itemPage.BindingContext = selectedItem;
            await Navigation.PushAsync(itemPage);
        }

        private async void CreateItem(object sender, EventArgs e)
        {
            Item item = new Item();
            ItemPage itemPage = new ItemPage();
            itemPage.BindingContext = item;
            await Navigation.PushAsync(itemPage);
        }
    }
}
```

Шаг 8

Запустим приложение.

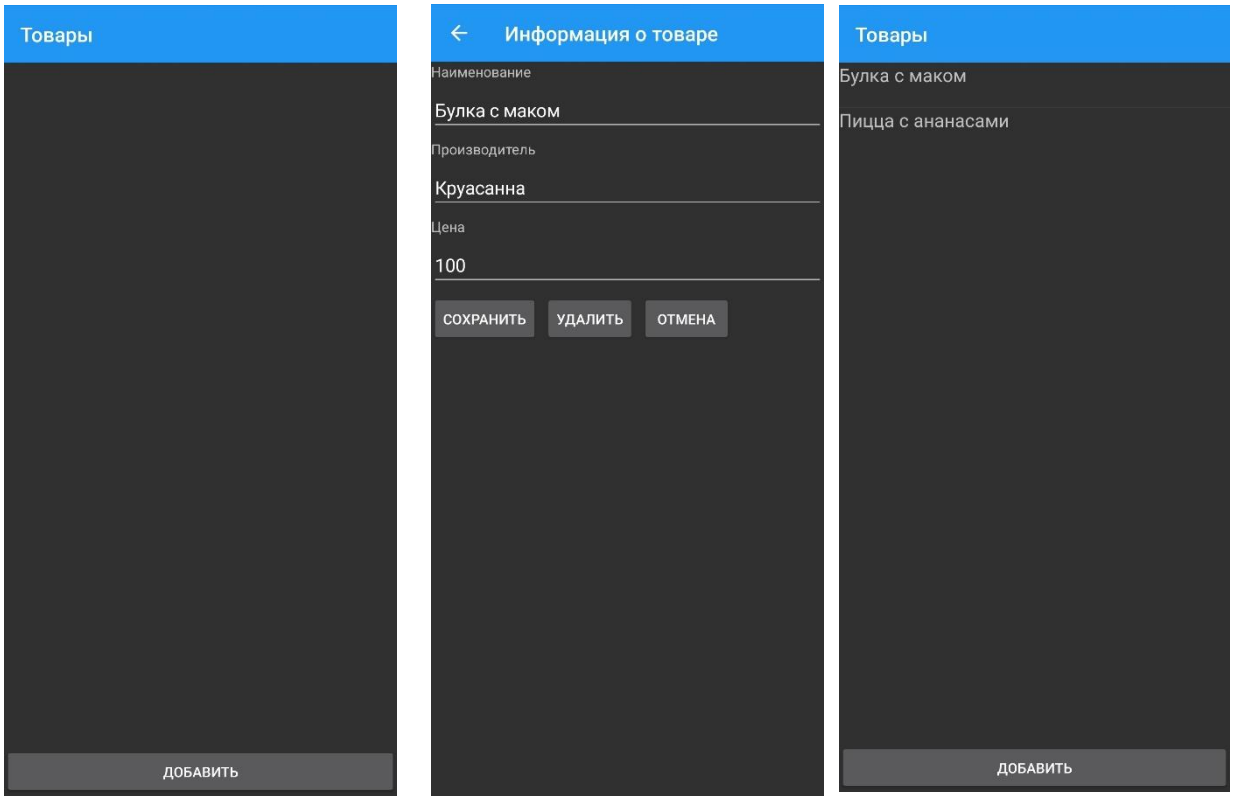

Рисунок 3 – Работающее приложение

Практическая часть

Разработать приложение для работы с БД Supply, которая хранит в себе информацию о товарах, поставщиках и поставках.

Содержание отчета

1. Титульный лист
2. Цель работы
3. Задание
4. Код программы
5. Результат выполнения