**🚀 Full Guide: QR Code Scanner App with SQL Server Connection in .NET MAUI**

We’ll go through **installing dependencies, setting up UI, writing code for QR scanning, and connecting to SQL Server**.

**📌 Step 1: Create a New .NET MAUI Project**

* **Open Visual Studio 2022**.
* Click **Create a new project**.
* Search for **.NET MAUI App** and select it.
* Name the project (e.g., QRCodeScannerApp) and choose a location.
* Click **Create**.

**📌 Step 2: Install Required Packages**

You'll need two NuGet packages:  
1️⃣ **ZXing.Net.Maui** → For QR Code scanning  
2️⃣ **Microsoft.Data.SqlClient** → For SQL Server connection

**Install via NuGet Package Manager**

* Open **Visual Studio**.
* Go to **Tools** → **NuGet Package Manager** → **Manage NuGet Packages for Solution**.
* Search for:
* ZXing.Net.Maui → Install
* Microsoft.Data.SqlClient → Install

**📌 Step 3: Configure MAUI to Use QR Scanner**

Modify **MauiProgram.cs** to enable barcode scanning.

**MauiProgram.cs**

using Microsoft.Extensions.Logging;

using ZXing.Net.Maui.Controls;

namespace QRCodeScannerApp;

public static class MauiProgram

{

public static MauiApp CreateMauiApp()

{

var builder = MauiApp.CreateBuilder();

builder

.UseMauiApp<App>()

.ConfigureFonts(fonts =>

{

fonts.AddFont("OpenSans-Regular.ttf", "OpenSansRegular");

fonts.AddFont("OpenSans-Semibold.ttf", "OpenSansSemibold");

})

.UseBarcodeReader();

#if DEBUG

builder.Logging.AddDebug();

#endif

return builder.Build();

}

}

**📌 Step 4: Create UI in XAML (MainPage.xaml)**

Modify **MainPage.xaml** to add:

* A **QR Code Scanner**.
* A **TextBox** to show the scanned result.
* A **Button** to send the result to the database.

**MainPage.xaml**

<?xml version="1.0" encoding="utf-8" ?>

<ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"

xmlns:zxing="clr-namespace:ZXing.Net.Maui.Controls;assembly=ZXing.Net.MAUI.Controls"

x:Class="QRCodeScannerApp.MainPage">

<ContentPage.Resources>

<Style x:Key="HeaderLabelStyle" TargetType="Label">

<Setter Property="FontSize" Value="24"/>

<Setter Property="HorizontalOptions" Value="Center"/>

</Style>

<Style x:Key="EntryStyle" TargetType="Entry">

<Setter Property="FontSize" Value="18"/>

<Setter Property="IsReadOnly" Value="True"/>

<Setter Property="Margin" Value="0,20,0,20"/>

<!-- Left, Top, Right, Bottom -->

</Style>

</ContentPage.Resources>

<ScrollView>

<VerticalStackLayout Padding="20">

<Label Text="Scan a QR Code" Style="{StaticResource HeaderLabelStyle}"

AutomationProperties.Name="Scan a QR Code"/>

<!-- QR Code Scanner -->

<!--<zxing:CameraBarcodeReaderView x:Name="cameraView"

BarcodesDetected="OnBarcodeDetected"

HeightRequest="300"

AutomationProperties.Name="QR Code Scanner"/>-->

<zxing:CameraBarcodeReaderView

x:Name="BarcodeReader"

BarcodesDetected="OnBarcodeDetected"/>

<!-- Display Scanned QR Code -->

<Entry x:Name="ScannedValue" Style="{StaticResource EntryStyle}"

Placeholder="Scanned QR Code"

AutomationProperties.Name="Scanned QR Code"/>

<!-- Button to Send Data to Database -->

<Button Text="Submit" Clicked="OnSendToDatabaseClicked"

AutomationProperties.Name="Send to Database Button"/>

</VerticalStackLayout>

</ScrollView>

</ContentPage>

**📌 Step 5: Handle QR Code Scanning in C#**

Modify **MainPage.xaml.cs** to handle the scanned QR code.

**MainPage.xaml.cs**

using Microsoft.Maui.Controls;

using ZXing.Net.Maui;

using Microsoft.Data.SqlClient;

using System.Threading.Tasks;

using ZXing.Net.Maui.Controls;

namespace QRCodeScannerApp

{

public partial class MainPage : ContentPage

{

// Replace with your actual SQL Server details

private string connectionString = "Data Source=ServerName;Database=TEST;Integrated Security=True;Trust Server Certificate=True;";

public MainPage()

{

InitializeComponent();

BarcodeReader.Options = new ZXing.Net.Maui.BarcodeReaderOptions

{

Formats = ZXing.Net.Maui.BarcodeFormat.QrCode,

AutoRotate = true,

Multiple = true

};

/\* cameraView.BarcodesDetected += OnBarcodeDetected; \*/ // Attach event handler

}

private void barcodeReader\_BarcodeDetected(object sender,ZXing.Net.Maui.BarcodeDetectionEventArgs e)

{

var first = e.Results?.FirstOrDefault();

if(first is null)

{

return;

}

Dispatcher.DispatchAsync(async () =>

{

await DisplayAlert("Barcode Detected", first.Value, "OK");

});

}

// Capture scanned QR Code and display in UI

private void OnBarcodeDetected(object sender, BarcodeDetectionEventArgs e)

{

MainThread.BeginInvokeOnMainThread(() =>

{

if (e.Results.Count() > 0)

{

ScannedValue.Text = e.Results[0].Value; // Display the first scanned result

}

});

}

// Send Scanned Value to Database

private async void OnSendToDatabaseClicked(object sender, EventArgs e)

{

if (string.IsNullOrWhiteSpace(ScannedValue.Text))

{

await DisplayAlert("Error", "No QR code scanned!", "OK");

return;

}

bool success = await SaveQrCodeToDatabase(ScannedValue.Text);

if (success)

await DisplayAlert("Success", "QR Code saved!", "OK");

else

await DisplayAlert("Error", "Failed to save QR Code.", "OK");

}

// Function to Connect to SQL Server and Insert Scanned Data

//private async Task<bool> SaveQrCodeToDatabase(string qrCodeValue)

//{

// try

// {

// using (SqlConnection conn = new SqlConnection(connectionString))

// {

// await conn.OpenAsync();

// string query = "INSERT INTO QRCodeTable (QrCodeValue, ScannedAt) VALUES (@QrValue, GETDATE())";

// using (SqlCommand cmd = new SqlCommand(query, conn))

// {

// cmd.Parameters.AddWithValue("@QrValue", qrCodeValue);

// int rows = await cmd.ExecuteNonQueryAsync();

// return rows > 0;

// }

// }

// }

// catch (Exception ex)

// {

// await DisplayAlert("Database Error", ex.Message, "OK");

// return false;

// }

//}

// Function to Update ScannedAt in SQL Server

private async Task<bool> SaveQrCodeToDatabase(string qrCodeValue)

{

try

{

using (SqlConnection conn = new SqlConnection(connectionString))

{

await conn.OpenAsync();

// Update the ScannedAt column where QrCodeValue matches

string query = "UPDATE QRCodeTable SET ScannedAt = GETDATE() WHERE QrCodeValue = @QrValue";

using (SqlCommand cmd = new SqlCommand(query, conn))

{

cmd.Parameters.AddWithValue("@QrValue", qrCodeValue);

int rows = await cmd.ExecuteNonQueryAsync();

return rows > 0; // Return true if at least one row is updated

}

}

}

catch (Exception ex)

{

await DisplayAlert("Database Error", ex.Message, "OK");

return false;

}

}

}

}

**📌 Step 6: Setup SQL Server**

Before running the app, create a table in **SQL Server**.

**Run this SQL Query in SQL Server Management Studio (SSMS)**

sql

CopyEdit

CREATE TABLE QRCodeTable (

Id INT IDENTITY(1,1) PRIMARY KEY,

QrCodeValue NVARCHAR(255),

ScannedAt DATETIME DEFAULT GETDATE()

);

ADD Login functionality

LoginPage.xaml

<?xml version="1.0" encoding="utf-8" ?>

<ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"

x:Class="QRCodeScannerApp.LoginPage"

Title="Login Page">

<VerticalStackLayout Padding="20" Margin="10">

<Label Text="Welcome..!" FontSize="24" HorizontalOptions="Center" Margin="0,0,0,20"/>

<Entry x:Name="UsernameEntry" Placeholder="Enter Username" Margin="0,0,0,10"/>

<Entry x:Name="PasswordEntry" Placeholder="Enter Password" IsPassword="True" Margin="0,0,0,10"/>

<Button Text="Login" Clicked="OnLoginClicked"/>

</VerticalStackLayout>

</ContentPage>

Steps to Implement Login in Your App

Modify Your SQL Server Database

sql

Copy

Edit

CREATE TABLE Users (

Id INT IDENTITY(1,1) PRIMARY KEY,

Username NVARCHAR(50) NOT NULL UNIQUE,

PasswordHash NVARCHAR(255) NOT NULL

);

2️⃣ Implement Login Logic (LoginPage.xaml.cs)

Authenticate users before allowing access to the QR code scanner.  
  
namespace QRCodeScannerApp;

using Microsoft.Data.SqlClient;

using Microsoft.Maui.Controls;

using System.Threading.Tasks;

public partial class LoginPage : ContentPage

{

private string connectionString = "Data Source=SERVER NAME;Database=TEST;Integrated Security=True;Trust Server Certificate=True;";

public LoginPage()

{

InitializeComponent();

}

private async void OnLoginClicked(object sender, EventArgs e)

{

string username = UsernameEntry.Text;

string password = PasswordEntry.Text;

if (string.IsNullOrWhiteSpace(username) || string.IsNullOrWhiteSpace(password))

{

await DisplayAlert("Error", "Please enter both username and password.", "OK");

return;

}

bool isAuthenticated = await AuthenticateUser(username, password);

if (isAuthenticated)

{

await DisplayAlert("Success", "Login successful!", "OK");

// Navigate to QR Scanner Page

await Navigation.PushAsync(new MainPage(username));

}

else

{

await DisplayAlert("Error", "Invalid credentials. Please try again.", "OK");

}

}

private async Task<bool> AuthenticateUser(string username, string password)

{

try

{

using (SqlConnection conn = new SqlConnection(connectionString))

{

await conn.OpenAsync();

string query = "SELECT COUNT(\*) FROM Users WHERE Username = @Username AND PasswordHash = @Password";

using (SqlCommand cmd = new SqlCommand(query, conn))

{

cmd.Parameters.AddWithValue("@Username", username);

cmd.Parameters.AddWithValue("@Password", password);

int userCount = (int)await cmd.ExecuteScalarAsync();

return userCount > 0;

}

}

}

catch (Exception ex)

{

await DisplayAlert("Database Error", ex.Message, "OK");

return false;

}

}

}

3️⃣ Modify App.xaml.cs to Set Login as First Page

namespace QRCodeScannerApp

{

public partial class App : Application

{

public App()

{

InitializeComponent();

MainPage = new NavigationPage(new LoginPage()); // Start with Login Page

}

}

}

4️⃣ Navigate to QR Scanner After Login (MainPage.xaml)

<?xml version="1.0" encoding="utf-8" ?>

<ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"

xmlns:zxing="clr-namespace:ZXing.Net.Maui.Controls;assembly=ZXing.Net.MAUI.Controls"

x:Class="QRCodeScannerApp.MainPage"

Title="QR Code Scanner">

<ContentPage.Resources>

<Style x:Key="HeaderLabelStyle" TargetType="Label">

<Setter Property="FontSize" Value="24"/>

<Setter Property="HorizontalOptions" Value="Center"/>

</Style>

<Style x:Key="EntryStyle" TargetType="Entry">

<Setter Property="FontSize" Value="18"/>

<Setter Property="IsReadOnly" Value="True"/>

<Setter Property="Margin" Value="0,20,0,20"/>

<!-- Left, Top, Right, Bottom -->

</Style>

<Style x:Key="SubmitButton" TargetType="Button">

<Setter Property="Margin" Value="0,0,0,20"/>

<!-- Left, Top, Right, Bottom -->

</Style>

</ContentPage.Resources>

<ScrollView>

<VerticalStackLayout Padding="20">

<Label Text="Scan a QR Code" Style="{StaticResource HeaderLabelStyle}"

AutomationProperties.Name="Scan a QR Code"/>

<!-- QR Code Scanner -->

<!--<zxing:CameraBarcodeReaderView x:Name="cameraView"

BarcodesDetected="OnBarcodeDetected"

HeightRequest="300"

AutomationProperties.Name="QR Code Scanner"/>-->

<zxing:CameraBarcodeReaderView

x:Name="BarcodeReader"

BarcodesDetected="OnBarcodeDetected"/>

<!-- Display Scanned QR Code -->

<Entry x:Name="ScannedValue" Style="{StaticResource EntryStyle}"

Placeholder="Scanned QR Code"

AutomationProperties.Name="Scanned QR Code"/>

<!-- Button to Send Data to Database -->

<Button Text="Submit" Clicked="OnSendToDatabaseClicked" Style="{StaticResource SubmitButton}"

AutomationProperties.Name="Send to Database Button"/>

<Button Text="Logout" Clicked="OnLogoutClicked" Style="{StaticResource LogoutButton}" BackgroundColor="Red" TextColor="White"/>

</VerticalStackLayout>

</ScrollView>

</ContentPage>

MainPage.xaml.cs  
  
using Microsoft.Maui.Controls;

using ZXing.Net.Maui;

using Microsoft.Data.SqlClient;

using System.Threading.Tasks;

using ZXing.Net.Maui.Controls;

namespace QRCodeScannerApp

{

public partial class MainPage : ContentPage

{

private string loggedInUser;

// Replace with your actual SQL Server details

private string connectionString = "Data Source=SERVER NAME;Database=TEST;Integrated Security=True;Trust Server Certificate=True;";

public MainPage(string username)

{

InitializeComponent();

loggedInUser = username; // Store the logged-in username

BarcodeReader.Options = new ZXing.Net.Maui.BarcodeReaderOptions

{

Formats = ZXing.Net.Maui.BarcodeFormat.QrCode,

AutoRotate = true,

Multiple = true

};

/\* cameraView.BarcodesDetected += OnBarcodeDetected; \*/ // Attach event handler

}

private void barcodeReader\_BarcodeDetected(object sender,ZXing.Net.Maui.BarcodeDetectionEventArgs e)

{

var first = e.Results?.FirstOrDefault();

if(first is null)

{

return;

}

Dispatcher.DispatchAsync(async () =>

{

await DisplayAlert("Barcode Detected", first.Value, "OK");

});

}

// Capture scanned QR Code and display in UI

private void OnBarcodeDetected(object sender, BarcodeDetectionEventArgs e)

{

MainThread.BeginInvokeOnMainThread(() =>

{

if (e.Results.Count() > 0)

{

ScannedValue.Text = e.Results[0].Value; // Display the first scanned result

}

});

}

// Send Scanned Value to Database

private async void OnSendToDatabaseClicked(object sender, EventArgs e)

{

if (string.IsNullOrWhiteSpace(ScannedValue.Text))

{

await DisplayAlert("Error", "No QR code scanned!", "OK");

return;

}

bool success = await SaveQrCodeToDatabase(ScannedValue.Text, loggedInUser);

if (success)

await DisplayAlert("Success", "QR Code saved!", "OK");

else

await DisplayAlert("Error", "Failed to save QR Code.", "OK");

}

// Function to Connect to SQL Server and Insert Scanned Data

//private async Task<bool> SaveQrCodeToDatabase(string qrCodeValue)

//{

// try

// {

// using (SqlConnection conn = new SqlConnection(connectionString))

// {

// await conn.OpenAsync();

// string query = "INSERT INTO QRCodeTable (QrCodeValue, ScannedAt) VALUES (@QrValue, GETDATE())";

// using (SqlCommand cmd = new SqlCommand(query, conn))

// {

// cmd.Parameters.AddWithValue("@QrValue", qrCodeValue);

// int rows = await cmd.ExecuteNonQueryAsync();

// return rows > 0;

// }

// }

// }

// catch (Exception ex)

// {

// await DisplayAlert("Database Error", ex.Message, "OK");

// return false;

// }

//}

// Function to Update ScannedAt in SQL Server

private async Task<bool> SaveQrCodeToDatabase(string qrCodeValue, string username)

{

try

{

using (SqlConnection conn = new SqlConnection(connectionString))

{

await conn.OpenAsync();

// Update the ScannedAt and ScannedBy columns where QrCodeValue matches

string query = "UPDATE QRCodeTable SET ScannedAt = GETDATE(), ScannedBy = @Username WHERE QrCodeValue = @QrValue";

using (SqlCommand cmd = new SqlCommand(query, conn))

{

cmd.Parameters.AddWithValue("@QrValue", qrCodeValue);

cmd.Parameters.AddWithValue("@Username", username); // Add username

int rows = await cmd.ExecuteNonQueryAsync();

return rows > 0; // Return true if at least one row is updated

}

}

}

catch (Exception ex)

{

await DisplayAlert("Database Error", ex.Message, "OK");

return false;

}

}

private async void OnLogoutClicked(object sender, EventArgs e)

{

bool confirm = await DisplayAlert("Logout", "Are you sure you want to log out?", "Yes", "No");

if (confirm)

{

// Navigate back to LoginPage and clear navigation history

Application.Current.MainPage = new NavigationPage(new LoginPage());

}

}

}

}  
  
3️⃣ Modify Your SQL Table (QRCodeTable)

Ensure your table has a column to store the username of the scanner.

sql

Copy

Edit

ALTER TABLE QRCodeTable ADD ScannedBy NVARCHAR(50);