using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Threading;

using System.Windows.Forms;

using FireSharp.Config;

using FireSharp.Response;

using FireSharp.Interfaces;

namespace GatewayExpress

{

public partial class Dashboard : Form

{

public static string visitPass;

MyUser c = new MyUser();

Thread th; //declaration of thread as th

int panelWidth;

bool Hidden;

int Counters = 0;

DateTime time = DateTime.Now;

int[] carCounter = new int[3] {0,0,0};

string[] carOwnerStatus = new string[3] { "Home Owners", "Visitors", "Passers" };

IFirebaseConfig ifc = new FirebaseConfig()

{

AuthSecret = "P2k1udVSBnRvxFKNk1Ve5OVZo8A4x90jWAEFXqYH",

BasePath = "https://lancaster-new-city-default-rtdb.firebaseio.com/"

};

IFirebaseClient client;

public Dashboard()

{

InitializeComponent();

panelWidth = PanelSlide.Width;

Hidden = false;

}

private void Dashboard\_Load(object sender, EventArgs e)

{

try

{

client = new FireSharp.FirebaseClient(ifc);

DataTable dt = new DataTable();

dt.Columns.Add("OR NUMBER");

dt.Columns.Add("USERNAME");

dt.Columns.Add("GIVEN NAME");

dt.Columns.Add("LAST NAME");

dt.Columns.Add("DATE");

dt.Columns.Add("TIME");

dt.Columns.Add("STATUS");

dt.Columns.Add("AMOUNT");

dt.Columns.Add("PURPOSE");

dataGridView1.Rows.Clear();

FirebaseResponse res = client.Get(@"COUNTER/COUNTER");

int Counter = int.Parse(res.ResultAs<String>());

for (int i = Counter; i > 0; i--)

{

FirebaseResponse res2 = client.Get(@"SERIAL NUMBER/" + i + "/OR\_NUMBER");

string RollNo = res2.ResultAs<string>();

var res3 = client.Get(@"TOLL/" + RollNo);

MyUser std = res3.ResultAs<MyUser>();

if (std.GIVEN\_NAME != "")//(std!=null)

{

dt.Rows.Add(std.OR\_NUMBER, std.USERNAME, std.GIVEN\_NAME, std.LAST\_NAME, std.DATE, std.TIME, std.STATUS, std.AMOUNT, std.PURPOSE);

if (time.ToString("MM/dd/yyyy") == std.DATE)

{

Counters++;

if (std.STATUS == "Home Owner" || std.STATUS == "Home Owner (Head)") { carCounter[0]++; }

else if (std.STATUS == "Visitor") { carCounter[1]++; }

else if (std.STATUS == "Passer") { carCounter[2]++; }

}

}

}

visitPass = carCounter[1].ToString();

foreach (DataRow item in dt.Rows)

{

dataGridView1.Rows.Add(item.ItemArray);

}

for (int i = 0; i < 3; i++)

{

if (carCounter[i] != 0)

{

chartEntry.Series["EntryP"].Points.AddXY(carOwnerStatus[i], carCounter[i]);

lblHomeOwners.Text = carCounter[0].ToString();

lblVisitors.Text = carCounter[1].ToString();

lblPassers.Text = carCounter[2].ToString();

}

}

lblCount.Text = Counters.ToString();

}

catch

{

//MessageBox.Show("No Internet or Connection Problem");

}

dataGridView1.BorderStyle = BorderStyle.None;

dataGridView1.AlternatingRowsDefaultCellStyle.BackColor = Color.FromArgb(238, 239, 249);

dataGridView1.CellBorderStyle = DataGridViewCellBorderStyle.SingleHorizontal;

dataGridView1.DefaultCellStyle.SelectionBackColor = Color.DarkTurquoise;

dataGridView1.DefaultCellStyle.SelectionForeColor = Color.WhiteSmoke;

dataGridView1.BackgroundColor = Color.White;

dataGridView1.EnableHeadersVisualStyles = false;

dataGridView1.ColumnHeadersBorderStyle = DataGridViewHeaderBorderStyle.None;

dataGridView1.ColumnHeadersDefaultCellStyle.BackColor = Color.FromArgb(36, 37, 45);

dataGridView1.ColumnHeadersDefaultCellStyle.ForeColor = Color.White;

}

private void timer1\_Tick(object sender, EventArgs e)

{

if (Hidden)

{

PanelSlide.Width = PanelSlide.Width + 10;

if (PanelSlide.Width >= panelWidth)

{

timer1.Stop();

Hidden = false;

this.Refresh();

}

}

else

{

PanelSlide.Width = PanelSlide.Width - 10;

if (PanelSlide.Width <= 0)

{

timer1.Stop();

Hidden = true;

this.Refresh();

}

}

}

private void openNewEToll(object obj)

{

Application.Run(new EToll()); //Run Etoll Tab

}

private void openNewLoad(object obj)

{

Application.Run(new Load()); //Run Load Tab

}

private void openNewForm1(object obj)

{

Application.Run(new Form1()); //Run Form1 or LoginForm Tab

}

private void openNewDrivers(object obj)

{

Application.Run(new Drivers()); //Run Drivers or User's Info Tab

}

private void openNewAdmins(object obj)

{

Application.Run(new Admins()); //Run Admin's Tab

}

private void btnDashboard\_Click\_1(object sender, EventArgs e)

{

this.Refresh();

}

private void btnToll\_Click\_1(object sender, EventArgs e)

{

try

{

this.Close(); //command for closing the application

th = new Thread(openNewEToll); //creating thread

th.SetApartmentState(ApartmentState.STA); //setting thread as STA

th.Start(); //start the thread

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

private void btnLoad\_Click\_1(object sender, EventArgs e)

{

this.Close(); //command for closing the application

th = new Thread(openNewLoad); //creating thread

th.SetApartmentState(ApartmentState.STA); //setting thread as STA

th.Start(); //start the thread

}

private void btnLogout\_Click\_1(object sender, EventArgs e)

{

this.Close(); //command for closing the application

th = new Thread(openNewForm1); //creating thread

th.SetApartmentState(ApartmentState.STA); //setting thread as STA

th.Start(); //start the thread

}

private void Dashboard\_FormClosing(object sender, FormClosingEventArgs e)

{

}

private void btnAdmins\_Click(object sender, EventArgs e)

{

this.Close(); //command for closing the application

th = new Thread(openNewAdmins); //creating thread

th.SetApartmentState(ApartmentState.STA); //setting thread as STA

th.Start(); //start the thread

}

private void btnDrivers\_Click(object sender, EventArgs e)

{

this.Close(); //command for closing the application

th = new Thread(openNewDrivers); //creating thread

th.SetApartmentState(ApartmentState.STA); //setting thread as STA

th.Start(); //start the thread

}

}

}