Hi-Tech Project

[Company name] | [Company address]

DBO.NET

ghazar ghazarian

2020

Contents

[Project Description: 2](#_Toc59088864)

[Project Design : 2](#_Toc59088865)

[Hitech DataBase Tables: 2](#_Toc59088866)

[Table’s Relations : 3](#_Toc59088867)

[Inter Face Designs : 3](#_Toc59088868)

[Log In To the system: 4](#_Toc59088869)

[MIS Management: 4](#_Toc59088870)

[Inventory Management: 5](#_Toc59088871)

[Order Management: 6](#_Toc59088872)

[Sales Management: 6](#_Toc59088873)

[Projects: 7](#_Toc59088874)

[Classes-project #1 Connected Mode: 7](#_Toc59088875)

[Classes-project #2 Entity Frame Work: 8](#_Toc59088876)

[Classes- Project#3 Disconnected mode: 8](#_Toc59088877)

[Project Implementation: 9](#_Toc59088878)

[Users – Employees: 9](#_Toc59088879)

[Sales management: 27](#_Toc59088880)

[Inventory control and Order management: 29](#_Toc59088881)

[Project Testing : 54](#_Toc59088882)

[Conclusion: 55](#_Toc59088883)

**HiTech Project**

# Project Description:

My project is about managing Order System that allows users to manage the company orders, inventory and even its employees, the system help the company to better control their sales records, human resources and their clients records. these operations are linked to a database server to keep truck of any updates on the records in the company and save it safely.

The project solution contains three projects. First project was done by using ADO.NET Connected mode which allows MIS manager to add/update/delete users and employees in the company. Second project is responsible on letting the sales manager to add/update/ delete customers records and it was made by using ADO.NET in Disconnected Mode. And finally, third project lets inventory controller and the order clerks to manage orders and the inventory in the company, this project was done using Entity Framework.

Each project contains three layers.

BLL: Contains all Business Classes

DAL: Contains all required code to manipulate the database and have access to the database.

GUI: contains the interface forms which allows users interact with the system and update the records in the database underseen.

# Project Design :

HiTech project contains multiple tables to keep truck of any action and save necessary records .

## Hitech DataBase Tables:

1. Users
2. Emplyees
3. Books
4. Categories
5. Jobs
6. Publishers
7. Authors
8. Author-Books
9. Orders
10. Customers

## Table’s Relations :

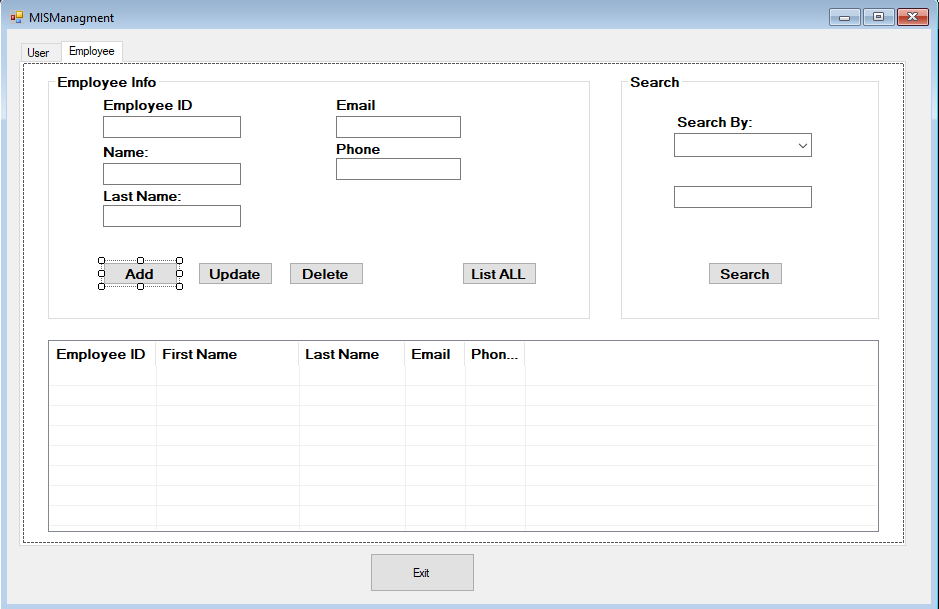


## Inter Face Designs :

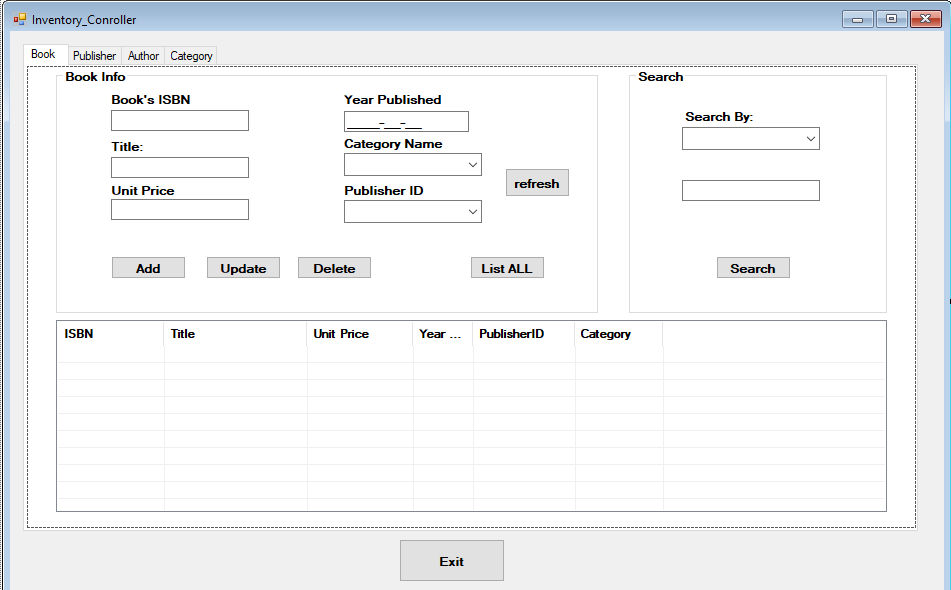
### Log In To the system:



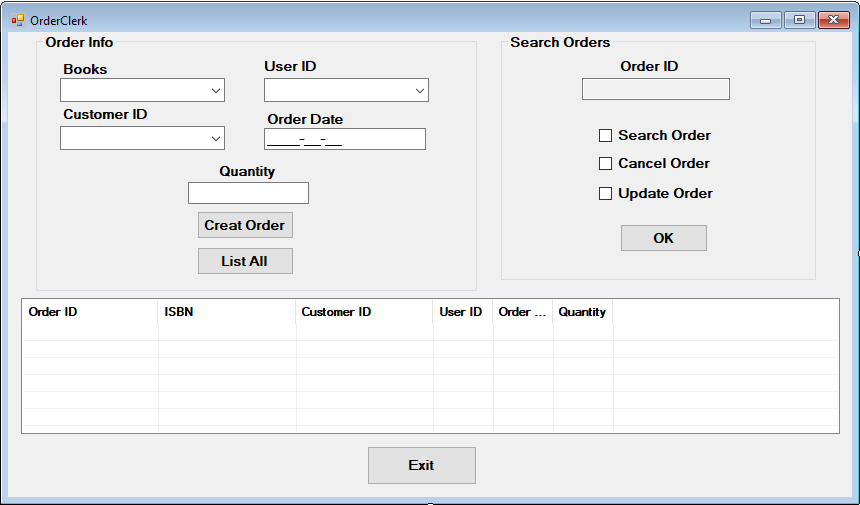
### MIS Management:



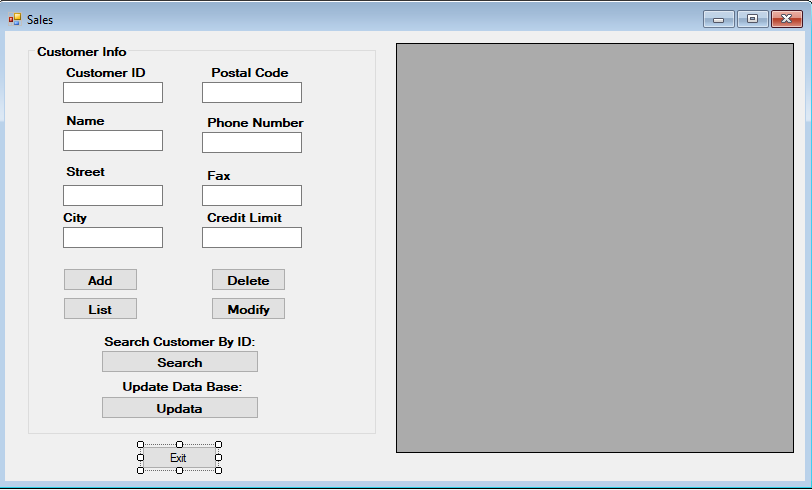
### Inventory Management:

****

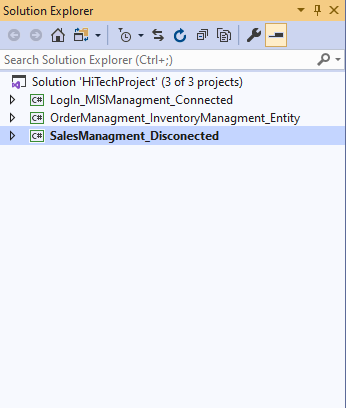
### Order Management:

****

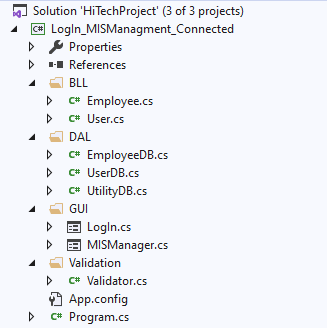
### Sales Management:

****

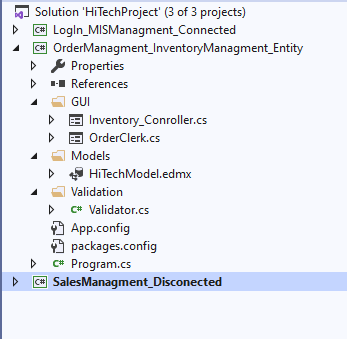
### Projects:



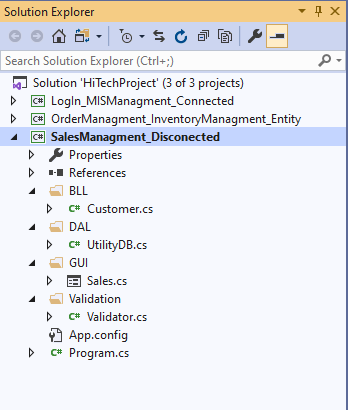
### Classes-project #1 Connected Mode:



### Classes-project #2 Entity Frame Work:



### Classes- Project#3 Disconnected mode:



# Project Implementation:

## Users – Employees:

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.Windows.Forms;

using HiTechProject.BLL;

using HiTechProject.DAL;

using HiTechProject.Validation;

namespace HiTechProject

{

public partial class Form1 : Form

{

int jobID;

public Form1()

{

InitializeComponent();

}

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

{

User userV = new User();

User userE = new User();

User user = new User();

string id = textBoxUserID.Text.Trim();

string name = textBoxUserFirstName.Text.Trim();

string email = textBoxUserEmail.Text.Trim();

string phone = textBoxUserPhone.Text.Trim();

if (!(Validator.IsValidName(name)) || !(Validator.IsEmpty(name)))

{

MessageBox.Show("wrong Name- must be charecters only", "error");

return;

}

if (!(Validator.IsValidEmail(email)) || !(Validator.IsEmpty(email)))

{

MessageBox.Show("wrong email adresse", "error");

return;

}

if (!(Validator.IsValidPhone(phone)) || !(Validator.IsEmpty(phone)))

{

MessageBox.Show("wrong phone- Must be 10 Digits", "error");

return;

}

if (!(Validator.IsValidId(id)) || !(Validator.IsEmpty(id)))

{

MessageBox.Show("wrong ID", "error");

return;

}

userV = userV.SearchUser(Convert.ToInt32(id));

if (userV != null)

{

MessageBox.Show("User already exist", "error");

return;

}

userE = userE.SearchEmpInUser(Convert.ToInt32(id));

if (userE != null)

{

MessageBox.Show("Employee already assigned", "error");

return;

}

if (comboBoxUserEmployeeID.SelectedIndex == -1)

{

MessageBox.Show("Please select the employee.", "Error");

return;

}

int employeeid = Convert.ToInt32(comboBoxUserEmployeeID.SelectedValue);

user.UserID = Convert.ToInt32(textBoxUserID.Text.Trim());

user.First\_Name = textBoxUserFirstName.Text.Trim();

user.Last\_Name = textBoxUserLastName.Text.Trim();

user.Email = textBoxUserEmail.Text.Trim();

user.Phone\_Number = textBoxUserPhone.Text.Trim();

user.Password = textBoxUserPassword.Text.Trim();

user.UserEmpID = employeeid;

user.JobID = jobID;

user.AddUser(user);

MessageBox.Show("User data saved successfully.", "Confirmation", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

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

{

string name = textBoxEmployeeFirstName.Text.Trim();

string email = textBoxEmployeeEmail.Text.Trim();

string phone = textBoxEmployeePhone.Text.Trim();

string id = textBoxEmployeeID.Text.Trim();

if (!(Validator.IsValidName(name)) || !(Validator.IsEmpty(name)))

{

MessageBox.Show("wrong Name- must be charecters only", "error");

return;

}

if (!(Validator.IsValidEmail(email)) || !(Validator.IsEmpty(email)))

{

MessageBox.Show("wrong email adresse", "error");

return;

}

if (!(Validator.IsValidPhone(phone)) || !(Validator.IsEmpty(phone)))

{

MessageBox.Show("wrong phone- Must be 10 Digits", "error");

return;

}

if (!(Validator.IsValidId(id)) || !(Validator.IsEmpty(id)))

{

MessageBox.Show("wrong ID", "error");

return;

}

Employee empid = new Employee();

empid = empid.SearchEmployee(Convert.ToInt32(id));

if (empid != null)

{

MessageBox.Show("Employee already exist", "error");

return;

}

Employee emp = new Employee();

emp.EmployeeID = Convert.ToInt32(textBoxEmployeeID.Text.Trim());

emp.First\_Name = textBoxEmployeeFirstName.Text.Trim();

emp.Last\_Name = textBoxEmployeeLastName.Text.Trim();

emp.Email = textBoxEmployeeEmail.Text.Trim();

emp.Phone\_Number = textBoxEmployeePhone.Text.Trim();

emp.SaveEmployee(emp);

MessageBox.Show("Employee data saved successfully.", "Confirmation", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

private void comboBoxUserJob\_SelectedIndexChanged\_1(object sender, EventArgs e)

{

int indexSelected = comboBoxUserJob.SelectedIndex;

switch (indexSelected)

{

case 0:

jobID = 1;

break;

case 1:

jobID = 2;

break;

case 2:

jobID = 3;

break;

case 3:

jobID = 4;

break;

default:

break;

}

}

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

{

Employee emp = new Employee();

List<Employee> listEmp = emp.ListEmployees();

listViewEmployees.Items.Clear();

if (listEmp.Count != 0)

{

foreach (Employee anEmp in listEmp)

{

ListViewItem item = new ListViewItem(anEmp.EmployeeID.ToString());

item.SubItems.Add(anEmp.First\_Name);

item.SubItems.Add(anEmp.Last\_Name);

item.SubItems.Add(anEmp.Email);

item.SubItems.Add(anEmp.Phone\_Number);

listViewEmployees.Items.Add(item);

}

}

else

{

MessageBox.Show("Empty List!", "No Employee");

}

}

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

{

DialogResult delete = MessageBox.Show("Deleting This Employee will remove all his/her records", "Confirmation", MessageBoxButtons.YesNo, MessageBoxIcon.Information);

if (delete == DialogResult.Yes)

{

Employee emp = new Employee();

int EmployeeID = Convert.ToInt32(textBoxEmployeeID.Text.Trim());

emp.DeleteEmployee(EmployeeID);

MessageBox.Show("Employee Succefully Deleted");

}

else

{

textBoxEmployeeID.Text = string.Empty;

MessageBox.Show("employee couldn't be found!", "Error");

}

}

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

{

DialogResult delete = MessageBox.Show("Do really want to delete this user?", "Confirmation", MessageBoxButtons.YesNo, MessageBoxIcon.Information);

if (delete == DialogResult.Yes)

{

User us = new User();

int userID = Convert.ToInt32(textBoxUserID.Text.Trim());

us.DeleteUser(userID);

MessageBox.Show("User Succefully Deleted");

}

else

{

textBoxEmployeeID.Text = string.Empty;

MessageBox.Show("User couldn't be found!", "Error");

}

}

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

{

User us = new User();

List<User> listusr = us.ListUsers();

listViewUsers.Items.Clear();

if (listusr.Count != 0)

{

foreach (User usr in listusr)

{

ListViewItem item = new ListViewItem(usr.UserID.ToString());

item.SubItems.Add(usr.First\_Name);

item.SubItems.Add(usr.Last\_Name);

item.SubItems.Add(usr.Phone\_Number);

item.SubItems.Add(usr.Email);

item.SubItems.Add(usr.JobID.ToString());

item.SubItems.Add(usr.UserEmpID.ToString());

item.SubItems.Add(usr.Password);

listViewUsers.Items.Add(item);

}

}

else

{

MessageBox.Show("User List!", "No User");

}

}

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

{

int indexSelected = comboBoxEmployeeSearch.SelectedIndex;

string id = textBoxUserSearch.Text.Trim();

if (!(Validator.IsValidId(id)) || !(Validator.IsEmpty(id)))

{

MessageBox.Show("wrong ID", "error");

return;

}

switch (indexSelected)

{

case 0:

User use = new User();

use.UserID = Convert.ToInt32(textBoxUserSearch.Text);

use = use.SearchUser (use.UserID);

listViewEmployees.Items.Clear();

if (use != null)

{

ListViewItem item = new ListViewItem(use.UserID.ToString());

item.SubItems.Add(use.First\_Name);

item.SubItems.Add(use.Last\_Name);

item.SubItems.Add(use.Email);

item.SubItems.Add(use.Phone\_Number);

item.SubItems.Add(use.Password);

item.SubItems.Add(use.JobID.ToString());

item.SubItems.Add(use.Phone\_Number);

listViewEmployees.Items.Add(item);

}

else

{

MessageBox.Show("User not found");

}

break;

case 1:

User use1 = new User();

List<User> listusr = use1.SearchUser(textBoxUserSearch.Text.ToString());

listViewEmployees.Items.Clear();

if (listusr.Count != 0)

{

foreach (User usr in listusr)

{

ListViewItem item = new ListViewItem(usr.UserID.ToString());

item.SubItems.Add(usr.First\_Name);

item.SubItems.Add(usr.Last\_Name);

item.SubItems.Add(usr.Email);

item.SubItems.Add(usr.Phone\_Number);

item.SubItems.Add(usr.Password);

item.SubItems.Add(usr.JobID.ToString());

item.SubItems.Add(usr.Phone\_Number);

listViewEmployees.Items.Add(item);

}

}

else

{

MessageBox.Show("User not found!", "Error");

}

break;

case 2:

User use2 = new User();

List<User> listusr2 = use2.SearchUser(textBoxUserSearch.Text.ToString());

listViewEmployees.Items.Clear();

if (listusr2.Count != 0)

{

foreach (User us in listusr2)

{

ListViewItem item = new ListViewItem(us.UserID.ToString());

item.SubItems.Add(us.First\_Name);

item.SubItems.Add(us.Last\_Name);

item.SubItems.Add(us.Email);

item.SubItems.Add(us.Phone\_Number);

item.SubItems.Add(us.Password);

item.SubItems.Add(us.JobID.ToString());

item.SubItems.Add(us.Phone\_Number);

listViewEmployees.Items.Add(item);

}

}

else

{

MessageBox.Show("User not found!", "Error");

}

break;

default:

break;

}

}

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

{

int indexSelected = comboBoxEmployeeSearch.SelectedIndex;

string id = textBoxEmployeeSearch.Text.Trim();

if (!(Validator.IsValidId(id)) || !(Validator.IsEmpty(id)))

{

MessageBox.Show("wrong ID", "error");

return;

}

switch (indexSelected)

{

case 0:

Employee emp = new Employee();

emp.EmployeeID = Convert.ToInt32(textBoxEmployeeSearch.Text);

emp = emp.SearchEmployee(emp.EmployeeID);

listViewEmployees.Items.Clear();

if (emp != null)

{

ListViewItem item = new ListViewItem(emp.EmployeeID.ToString());

item.SubItems.Add(emp.First\_Name);

item.SubItems.Add(emp.Last\_Name);

item.SubItems.Add(emp.Email);

item.SubItems.Add(emp.Phone\_Number);

listViewEmployees.Items.Add(item);

}

else

{

MessageBox.Show("Employee not found");

}

break;

case 1:

Employee emp1 = new Employee();

List<Employee> listEmp = emp1.SearchEmployee(textBoxEmployeeSearch.Text.ToString());

listViewEmployees.Items.Clear();

if (listEmp.Count != 0)

{

foreach (Employee anEmp in listEmp)

{

ListViewItem item = new ListViewItem(anEmp.EmployeeID.ToString());

item.SubItems.Add(anEmp.First\_Name);

item.SubItems.Add(anEmp.Last\_Name);

item.SubItems.Add(anEmp.Email);

item.SubItems.Add(anEmp.Phone\_Number);

listViewEmployees.Items.Add(item);

}

}

else

{

MessageBox.Show("Employee not found!", "Error");

}

break;

case 2:

Employee emp2 = new Employee();

List<Employee> listEmp2 = emp2.SearchEmployee(textBoxEmployeeSearch.Text.ToString());

listViewEmployees.Items.Clear();

if (listEmp2.Count != 0)

{

foreach (Employee anEmp in listEmp2)

{

ListViewItem item = new ListViewItem(anEmp.EmployeeID.ToString());

item.SubItems.Add(anEmp.First\_Name);

item.SubItems.Add(anEmp.Last\_Name);

item.SubItems.Add(anEmp.Email);

item.SubItems.Add(anEmp.Phone\_Number);

listViewEmployees.Items.Add(item);

}

}

else

{

MessageBox.Show("Employee not found!", "Error");

}

break;

default:

break;

}

}

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

{

Employee emp = new Employee();

emp.EmployeeID = Convert.ToInt32(textBoxEmployeeID.Text.Trim());

emp.First\_Name = textBoxEmployeeFirstName.Text.Trim();

emp.Last\_Name = textBoxEmployeeLastName.Text.Trim();

emp.Email = textBoxEmployeeEmail.Text.Trim();

emp.Phone\_Number = textBoxEmployeePhone.Text.Trim();

DialogResult answer = MessageBox.Show("Do you really want to update this employee info? ", "Confirmation", MessageBoxButtons.YesNo, MessageBoxIcon.Information);

if (answer == DialogResult.Yes)

{

emp.UpdateEmployee(emp);

}

}

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

{

User usr = new User();

usr.UserEmpID = Convert.ToInt32(comboBoxUserEmployeeID.SelectedValue);

usr.First\_Name = textBoxUserFirstName.Text.Trim();

usr.Last\_Name = textBoxUserLastName.Text.Trim();

usr.Password = textBoxUserPassword.Text.Trim();

usr.Phone\_Number = textBoxUserPhone.Text.Trim();

usr.Email= textBoxUserEmail.Text.Trim();

usr.JobID = Convert.ToInt32(jobID);

usr.UserID = Convert.ToInt32(textBoxUserID.Text.Trim());

DialogResult answer = MessageBox.Show("Do you really want to update this User info? ", "Confirmation", MessageBoxButtons.YesNo, MessageBoxIcon.Information);

if (answer == DialogResult.Yes)

{

usr.UpdateUser(usr);

}

}

private void Form1\_Load(object sender, EventArgs e)

{

Employee emp = new Employee();

List<Employee> listEmp4 = emp.ListEmployees();

foreach (Employee emp4 in listEmp4)

{

comboBoxUserEmployeeID.DisplayMember = "First\_Name";

comboBoxUserEmployeeID.ValueMember = "EmployeeID";

comboBoxUserEmployeeID.DataSource = listEmp4;

}

}

private void comboBoxUserEmployeeID\_SelectedIndexChanged(object sender, EventArgs e)

{

}

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

{

Employee emp = new Employee();

List<Employee> listEmp4 = emp.ListEmployees();

foreach (Employee emp4 in listEmp4)

{

comboBoxUserEmployeeID.DisplayMember = "First\_Name";

comboBoxUserEmployeeID.ValueMember = "EmployeeID";

comboBoxUserEmployeeID.DataSource = listEmp4;

}

}

private void button1\_Click(object sender, EventArgs e)

{

DialogResult exit = MessageBox.Show("Do you really want to exit the application?", "Confirmation", MessageBoxButtons.YesNo, MessageBoxIcon.Information);

if (exit == DialogResult.Yes)

{

Application.Exit();

}

}

private void button2\_Click(object sender, EventArgs e)

{

int id = 1;

User ff = new User();

ff = ff.SearchEmpInUser(id);

MessageBox.Show(ff.Password);

}

}

}

## Sales management:

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.Windows.Forms;

using System.Data.SqlClient;

using SalesManagment;

namespace SalesManagment.GUI

{

public partial class Sales : Form

{

SqlDataAdapter da;

DataSet dsHiTech;

DataTable dtCustomers;

SqlCommandBuilder sqlBuilder;

public Sales()

{

InitializeComponent();

}

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

{

dsHiTech = new DataSet("HiTechDS");

dtCustomers = new DataTable("Customers");

dsHiTech.Tables.Add(dtCustomers);

dtCustomers.Columns.Add("CustomerID", typeof(Int32));

dtCustomers.Columns.Add("Name", typeof(string));

dtCustomers.Columns.Add("Street", typeof(string));

dtCustomers.Columns.Add("City", typeof(string));

dtCustomers.Columns.Add("Postal\_Code", typeof(string));

dtCustomers.Columns.Add("Phone\_Number", typeof(string));

dtCustomers.Columns.Add("Fax\_Number", typeof(string));

dtCustomers.Columns.Add("Credit\_Limit", typeof(int));

dtCustomers.Columns["CustomerID"].AutoIncrement = true;

dtCustomers.Columns["CustomerID"].AutoIncrementStep = 1;

dtCustomers.Columns["CustomerID"].AutoIncrementSeed = 1;

dtCustomers.PrimaryKey = new DataColumn[] { dtCustomers.Columns["CustomerID"] };

da = new SqlDataAdapter("SELECT \* FROM Customers", UtilityDB.ConnectDB());

sqlBuilder = new SqlCommandBuilder(da);

da.Fill(dsHiTech.Tables["Customers"]);

}

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

{

DataRow dr = dtCustomers.NewRow();

dr["Name"] = textBoxCustomerName.Text.Trim();

dr["Street"] = textBoxCustomerstreet.Text.Trim();

dr["City"] = textBoxCustomerCity.Text.Trim();

dr["Postal\_Code"] = textBoxCustomerPostalCode.Text.Trim();

dr["Phone\_Number"] = textBoxCustomerPhone.Text.Trim();

dr["Fax\_Number"] = textBoxCustomerFax.Text.Trim();

dr["Credit\_Limit"] = Convert.ToInt32(textBoxCustomerCreditLimit.Text.Trim());

dtCustomers.Rows.Add(dr);

//da.Fill(dsHiTech.Tables["Customers"]);

MessageBox.Show(dr.RowState.ToString());

}

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

{

int searchid = Convert.ToInt32(textBoxCustomerID.Text.Trim());

DataRow drCustomer = dtCustomers.Rows.Find(searchid);

if (drCustomer != null)

{

textBoxCustomerName.Text = drCustomer["Name"].ToString();

textBoxCustomerstreet.Text = drCustomer["Street"].ToString();

textBoxCustomerCity.Text = drCustomer["City"].ToString();

textBoxCustomerPostalCode.Text = drCustomer["Postal\_Code"].ToString();

textBoxCustomerPhone.Text = drCustomer["Phone\_Number"].ToString();

textBoxCustomerFax.Text = drCustomer["Fax\_Number"].ToString();

textBoxCustomerCreditLimit.Text = drCustomer["Credit\_limit"].ToString();

}

else

{

MessageBox.Show("Customer not found!", "Invalid Customer ID", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

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

{

dataGridViewCustomersDS.DataSource = dsHiTech.Tables["Customers"];

}

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

{

da.Update(dsHiTech.Tables["Customers"]);

MessageBox.Show("Database has been updated sucessfully.", "Confirmation");

}

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

{

int searchid = Convert.ToInt32(textBoxCustomerID.Text.Trim());

DataRow drCustomer = dtCustomers.Rows.Find(searchid);

if (drCustomer != null)

{

drCustomer["Name"] = textBoxCustomerName.Text.Trim();

drCustomer["Street"] = textBoxCustomerstreet.Text.Trim();

drCustomer["City"] = textBoxCustomerCity.Text.Trim();

drCustomer["Postal\_Code"] = textBoxCustomerPostalCode.Text.Trim();

drCustomer["Phone\_Number"] = textBoxCustomerPhone.Text.Trim();

drCustomer["Fax\_Number"] = textBoxCustomerFax.Text.Trim();

drCustomer["Credit\_Limit"] = Convert.ToInt32(textBoxCustomerCreditLimit.Text.Trim());

MessageBox.Show("Customer with ID: #"+ searchid + " Successfully Modified ", "Comfirmation", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

else

{

MessageBox.Show("Customer not found!", "Invalid Customer ID", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

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

{

int searchid = Convert.ToInt32(textBoxCustomerID.Text.Trim());

DataRow drCustomer = dtCustomers.Rows.Find(searchid);

if (drCustomer != null)

{

drCustomer.Delete();

MessageBox.Show("Customer with ID: #" + searchid + " Successfully Deleted ", "Comfirmation", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

else

{

MessageBox.Show("Customer not found!", "Invalid Customer ID", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

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

{

DialogResult exit = MessageBox.Show("Do you really want to exit the application?", "Confirmation", MessageBoxButtons.YesNo, MessageBoxIcon.Information);

if (exit == DialogResult.Yes)

{

Application.Exit();

}

}

}

}

## Inventory control and Order management:

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.Windows.Forms;

using OrderManagment\_InventoryManagment\_Entity.Models;

namespace OrderManagment.GUI

{

public partial class Inventory\_Conroller : Form

{

Hi\_TechDBEntities1 dBEntities = new Hi\_TechDBEntities1();

public Inventory\_Conroller()

{

InitializeComponent();

}

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

{

if (comboBoxBookCategoryName.SelectedIndex == -1)

{

MessageBox.Show("Please select a Category", "Error");

return;

}

else if (comboBoxBookPublisherID.SelectedIndex == -1)

{

MessageBox.Show("Please select a publisher", "Error");

return;

}

int publisherid = Convert.ToInt32(comboBoxBookPublisherID.SelectedValue);

int categoryid = Convert.ToInt32(comboBoxBookCategoryName.SelectedValue);

Book book = new Book();

book.ISBN = textBoxBookISBN.Text.Trim();

book.Title = textBoxBookTitle.Text.Trim();

book.UnitPrice = Convert.ToInt32(textBoxBookUnitPrice.Text.Trim());

book.YearPublished = Convert.ToDateTime(maskedTextBoxYearPublished.Text.Trim());

book.CategoryID = categoryid;

book.PublisherID = publisherid;

dBEntities.Books.Add(book);

dBEntities.SaveChanges();

MessageBox.Show("Book Added successfully.", "Confirmation");

}

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

{

if (comboBoxBookCategoryName.SelectedIndex == -1)

{

MessageBox.Show("Please select a Category", "Error");

return;

}

else if (comboBoxBookPublisherID.SelectedIndex == -1)

{

MessageBox.Show("Please select a publisher", "Error");

return;

}

int publisherid = Convert.ToInt32(comboBoxBookPublisherID.SelectedValue);

int categoryid = Convert.ToInt32(comboBoxBookCategoryName.SelectedValue);

var update = (from b in dBEntities.Books

where b.ISBN == textBoxBookISBN.Text.Trim()

select b).FirstOrDefault();

if (update != null)

{

update.ISBN = textBoxBookISBN.Text.Trim();

update.Title = textBoxBookTitle.Text.Trim();

update.UnitPrice = Convert.ToInt32(textBoxBookUnitPrice.Text.Trim());

update.YearPublished = Convert.ToDateTime(maskedTextBoxYearPublished.Text.Trim());

update.CategoryID = categoryid;

update.PublisherID = publisherid;

dBEntities.SaveChangesAsync();

}

else

{

MessageBox.Show("Wrong ISBN Number!", "Error");

}

}

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

{

int indexSelected = comboBoxBookSearch.SelectedIndex;

switch (indexSelected)

{

case 0:

var searchbookid = (from b in dBEntities.Books

where b.ISBN == textBoxBookSearch.Text.Trim()

select b).FirstOrDefault();

if (searchbookid != null)

{

listViewBooks.Items.Clear();

ListViewItem item = new ListViewItem(searchbookid.ISBN);

item.SubItems.Add(searchbookid.Title);

item.SubItems.Add(searchbookid.UnitPrice.ToString());

item.SubItems.Add(searchbookid.YearPublished.ToString());

item.SubItems.Add(searchbookid.CategoryID.ToString());

item.SubItems.Add(searchbookid.PublisherID.ToString());

listViewBooks.Items.Add(item);

}

else

{

MessageBox.Show("No book with that ISBN!", "Error");

}

break;

case 1:

var listname = (from b in dBEntities.Books

where b.Title == textBoxBookSearch.Text.Trim()

select b).ToList();

if (listname.Count > 0)

{

listViewBooks.Items.Clear();

foreach (var book in listname)

{

ListViewItem item = new ListViewItem(book.ISBN);

item.SubItems.Add(book.Title);

item.SubItems.Add(book.UnitPrice.ToString());

item.SubItems.Add(book.YearPublished.ToString());

item.SubItems.Add(book.CategoryID.ToString());

item.SubItems.Add(book.PublisherID.ToString());

listViewBooks.Items.Add(item);

}

}

else

{

MessageBox.Show("No book with that title!", "Error");

}

break;

case 2:

int pid = Convert.ToInt32(textBoxBookSearch.Text.Trim());

var searchPubid = (from b in dBEntities.Books

where b.PublisherID == pid

select b).FirstOrDefault();

if (searchPubid != null)

{

listViewBooks.Items.Clear();

ListViewItem item = new ListViewItem(searchPubid.ISBN);

item.SubItems.Add(searchPubid.Title);

item.SubItems.Add(searchPubid.UnitPrice.ToString());

item.SubItems.Add(searchPubid.YearPublished.ToString());

item.SubItems.Add(searchPubid.CategoryID.ToString());

item.SubItems.Add(searchPubid.PublisherID.ToString());

listViewBooks.Items.Add(item);

}

else

{

MessageBox.Show("No book assigned to that publisher ID!", "Error");

}

break;

}

}

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

{

var listbooks = dBEntities.Books.ToList();

listViewBooks.Items.Clear();

if (listbooks.Count != 0)

{

foreach (var book in listbooks)

{

ListViewItem item = new ListViewItem(book.ISBN);

item.SubItems.Add(book.Title);

item.SubItems.Add(book.UnitPrice.ToString());

item.SubItems.Add(book.YearPublished.ToString());

item.SubItems.Add(book.CategoryID.ToString());

item.SubItems.Add(book.PublisherID.ToString());

listViewBooks.Items.Add(item);

}

}

else

{

MessageBox.Show("books List!", "No book");

}

}

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

{

DialogResult answer = MessageBox.Show("Do really want to delete this book?", "Confirmation", MessageBoxButtons.YesNo, MessageBoxIcon.Information);

if (answer == DialogResult.Yes)

{

var delete = (from b in dBEntities.Books

where b.ISBN == textBoxBookISBN.Text.Trim()

select b).FirstOrDefault();

if (delete != null)

{

dBEntities.Books.Remove(delete);

dBEntities.SaveChanges();

}

else

{

MessageBox.Show("Wrong ISBN Number! or Book already Deleted", "No book");

}

}

else

{

textBoxBookISBN.Text = string.Empty;

return;

}

}

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

{

Publisher pub = new Publisher();

pub.PublisherID = Convert.ToInt32(textBoxPublisherID.Text.Trim());

pub.Name = textBoxPublisherName.Text.Trim();

pub.Phone\_Number = textBoxPublisherPhoneNumber.Text.Trim();

dBEntities.Publishers.Add(pub);

dBEntities.SaveChanges();

MessageBox.Show("Publisher Added successfully.", "Confirmation");

}

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

{

int pubid = Convert.ToInt32(textBoxPublisherID.Text.Trim());

var update = (from p in dBEntities.Publishers

where p.PublisherID == pubid

select p).FirstOrDefault();

if (update != null)

{

update.PublisherID = Convert.ToInt32(textBoxPublisherID.Text.Trim());

update.Name = textBoxPublisherName.Text.Trim();

update.Phone\_Number = textBoxPublisherPhoneNumber.Text.Trim();

dBEntities.SaveChangesAsync();

MessageBox.Show("Sussecfully Updated!", "Confirm");

}

else

{

MessageBox.Show("Wrong PublisherID Number!", "Error");

}

}

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

{

DialogResult answer = MessageBox.Show("Do really want to delete this Publisher?", "Confirmation", MessageBoxButtons.YesNo, MessageBoxIcon.Information);

if (answer == DialogResult.Yes)

{

int pubid = Convert.ToInt32(textBoxPublisherID.Text.Trim());

var delete = (from p in dBEntities.Publishers

where p.PublisherID == pubid

select p).FirstOrDefault();

if (delete != null)

{

dBEntities.Publishers.Remove(delete);

dBEntities.SaveChanges();

}

else

{

MessageBox.Show("Wrong ID Number! or Publisher already Deleted", "No Publisher");

}

}

else

{

textBoxPublisherID.Text = string.Empty;

return;

}

}

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

{

var listPublishers = dBEntities.Publishers.ToList();

listViewPublishers.Items.Clear();

if (listPublishers.Count != 0)

{

foreach (var pub in listPublishers)

{

ListViewItem item = new ListViewItem(pub.PublisherID.ToString());

item.SubItems.Add(pub.Name);

item.SubItems.Add(pub.Phone\_Number);

listViewPublishers.Items.Add(item);

}

}

else

{

MessageBox.Show("Publishers List!", "No Publishers");

}

}

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

{

int indexSelected = comboBoxPublisherSearch.SelectedIndex;

switch (indexSelected)

{

case 0:

int pubid = Convert.ToInt32(textBoxPublisherSearch.Text);

var searchpubid = (from p in dBEntities.Publishers

where p.PublisherID == pubid

select p).FirstOrDefault();

if (searchpubid != null)

{

listViewPublishers.Items.Clear();

ListViewItem item = new ListViewItem(searchpubid.PublisherID.ToString());

item.SubItems.Add(searchpubid.Name);

item.SubItems.Add(searchpubid.Phone\_Number);

listViewPublishers.Items.Add(item);

}

else

{

MessageBox.Show("No Publisher with that ID!", "Error");

}

break;

case 1:

var listname = (from p in dBEntities.Publishers

where p.Name == textBoxPublisherSearch.Text.Trim()

select p).ToList();

if (listname.Count > 0)

{

foreach (var pub in listname)

{

listViewPublishers.Items.Clear();

ListViewItem item = new ListViewItem(pub.PublisherID.ToString());

item.SubItems.Add(pub.Name);

item.SubItems.Add(pub.Phone\_Number);

listViewPublishers.Items.Add(item);

}

}

else

{

MessageBox.Show("No Publisher with that name!", "Error");

}

break;

}

}

private void Inventory\_Conroller\_Load(object sender, EventArgs e)

{

var listCtg = (from ctg in dBEntities.Categories

select ctg).ToList<Category>();

foreach (Category ctg in listCtg)

{

comboBoxBookCategoryName.DisplayMember = "Title";

comboBoxBookCategoryName.ValueMember = "CategoryID";

comboBoxBookCategoryName.DataSource = listCtg;

}

var listPblsh = (from pub in dBEntities.Publishers

select pub).ToList<Publisher>();

foreach (Publisher pub in listPblsh)

{

comboBoxBookPublisherID.DisplayMember = "Name";

comboBoxBookPublisherID.ValueMember = "PublisherID";

comboBoxBookPublisherID.DataSource = listPblsh;

}

}

private void button2\_Click(object sender, EventArgs e)

{

var listCtg = (from ctg in dBEntities.Categories

select ctg).ToList<Category>();

foreach (Category ctg in listCtg)

{

comboBoxBookCategoryName.DisplayMember = "Title";

comboBoxBookCategoryName.ValueMember = "CategoryID";

comboBoxBookCategoryName.DataSource = listCtg;

}

var listPblsh = (from pub in dBEntities.Publishers

select pub).ToList<Publisher>();

foreach (Publisher pub in listPblsh)

{

comboBoxBookPublisherID.DisplayMember = "Name";

comboBoxBookPublisherID.ValueMember = "PublisherID";

comboBoxBookPublisherID.DataSource = listPblsh;

}

}

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

{

Author auth = new Author();

auth.AuthorID = Convert.ToInt32(textBoxAuthorID.Text.Trim());

auth.First\_Name = textBoxAuthorFirstName.Text.Trim();

auth.Last\_Name = textBoxAuthorLastName.Text.Trim();

auth.Email = textBoxAuthorEmail.Text.Trim();

dBEntities.Authors.Add(auth);

dBEntities.SaveChanges();

MessageBox.Show("Author Added successfully.", "Confirmation");

}

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

{

int authid = Convert.ToInt32(textBoxAuthorID.Text.Trim());

var update = (from a in dBEntities.Authors

where a.AuthorID == authid

select a).FirstOrDefault();

if (update != null)

{

update.AuthorID = Convert.ToInt32(textBoxAuthorID.Text.Trim());

update.First\_Name = textBoxAuthorFirstName.Text.Trim();

update.Last\_Name = textBoxAuthorLastName.Text.Trim();

update.Email = textBoxAuthorEmail.Text.Trim();

dBEntities.SaveChangesAsync();

}

else

{

MessageBox.Show("Wrong Author ID Number!", "Error");

}

}

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

{

DialogResult answer = MessageBox.Show("Do really want to delete this Author?", "Confirmation", MessageBoxButtons.YesNo, MessageBoxIcon.Information);

if (answer == DialogResult.Yes)

{

int authrid = Convert.ToInt32(textBoxAuthorID.Text.Trim());

var delete = (from a in dBEntities.Authors

where a.AuthorID == authrid

select a).FirstOrDefault();

if (delete != null)

{

dBEntities.Authors.Remove(delete);

dBEntities.SaveChanges();

}

else

{

MessageBox.Show("Wrong Author ID or Author already Deleted", "No Author");

}

}

else

{

textBoxAuthorID.Text = string.Empty;

return;

}

}

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

{

listViewAuthors.Items.Clear();

var listauthors = dBEntities.Authors.ToList();

if (listauthors.Count != 0)

{

foreach (var auth in listauthors)

{

ListViewItem item = new ListViewItem(auth.AuthorID.ToString());

item.SubItems.Add(auth.First\_Name);

item.SubItems.Add(auth.Last\_Name);

item.SubItems.Add(auth.Email);

listViewAuthors.Items.Add(item);

}

}

else

{

MessageBox.Show("Authors List!", "No Author");

}

}

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

{

int indexSelected = comboBoxAuthorSearch.SelectedIndex;

switch (indexSelected)

{

case 0:

int authid = Convert.ToInt32(textBoxAuthorSearch.Text.Trim());

var searchAuthid = (from a in dBEntities.Authors

where a.AuthorID == authid

select a).FirstOrDefault();

if (searchAuthid != null)

{

listViewAuthors.Items.Clear();

ListViewItem item = new ListViewItem(searchAuthid.AuthorID.ToString());

item.SubItems.Add(searchAuthid.First\_Name);

item.SubItems.Add(searchAuthid.Last\_Name);

item.SubItems.Add(searchAuthid.Email);

listViewAuthors.Items.Add(item);

}

else

{

MessageBox.Show("Author ID Not Found!", "Error");

}

break;

case 1:

var listname = (from a in dBEntities.Authors

where a.First\_Name == textBoxAuthorSearch.Text.Trim()

select a).ToList();

if (listname.Count != 0)

{

listViewAuthors.Items.Clear();

foreach (var auth in listname)

{

ListViewItem item = new ListViewItem(auth.AuthorID.ToString());

item.SubItems.Add(auth.First\_Name);

item.SubItems.Add(auth.Last\_Name);

item.SubItems.Add(auth.Email);

listViewAuthors.Items.Add(item);

}

}

else

{

MessageBox.Show("Author Name Not Found!", "Error");

}

break;

}

}

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

{

Category cat = new Category();

cat.CategoryID = Convert.ToInt32(textBoxCategoryID.Text.Trim());

cat.Title = textBoxCategoryTitle.Text.Trim();

dBEntities.Categories.Add(cat);

dBEntities.SaveChanges();

MessageBox.Show("Successfuly Added", "Confirm");

}

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

{

int catid = Convert.ToInt32(textBoxCategoryID.Text.Trim());

var searchCatid = (from c in dBEntities.Categories

where c.CategoryID == catid

select c).FirstOrDefault();

if (searchCatid != null)

{

searchCatid.CategoryID = Convert.ToInt32(textBoxCategoryID.Text.Trim());

searchCatid.Title = textBoxCategoryTitle.Text.Trim();

dBEntities.SaveChanges();

MessageBox.Show("Successfuly updated", "Confirm");

}

else

{

MessageBox.Show("Wrong Category ID", "No Category");

}

}

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

{

DialogResult answer = MessageBox.Show("Do really want to delete this Category?", "Confirmation", MessageBoxButtons.YesNo, MessageBoxIcon.Information);

if (answer == DialogResult.Yes)

{

int catid = Convert.ToInt32(textBoxCategoryID.Text.Trim());

var delete = (from c in dBEntities.Categories

where c.CategoryID == catid

select c).FirstOrDefault();

if (delete != null)

{

dBEntities.Categories.Remove(delete);

dBEntities.SaveChanges();

}

else

{

MessageBox.Show("Wrong Category ID or Category already Deleted", "No Category");

}

}

else

{

textBoxCategoryID.Text = string.Empty;

return;

}

}

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

{

listViewCategories.Items.Clear();

var listcat = dBEntities.Categories.ToList();

if (listcat.Count != 0)

{

foreach (var cat in listcat)

{

ListViewItem item = new ListViewItem(cat.CategoryID.ToString());

item.SubItems.Add(cat.Title);

listViewCategories.Items.Add(item);

}

}

else

{

MessageBox.Show("Category List!", "No Category");

}

}

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

{

int indexSelected = comboBoxCategorySearch.SelectedIndex;

switch (indexSelected)

{

case 0:

int catid = Convert.ToInt32(textBoxCategorySearch.Text.Trim());

var searchCatid = (from c in dBEntities.Categories

where c.CategoryID == catid

select c).FirstOrDefault();

if (searchCatid != null)

{

listViewCategories.Items.Clear();

ListViewItem item = new ListViewItem(searchCatid.CategoryID.ToString());

item.SubItems.Add(searchCatid.Title);

listViewCategories.Items.Add(item);

}

else

{

MessageBox.Show("Category ID Not Found!", "Error");

}

break;

case 1:

var listname = (from a in dBEntities.Categories

where a.Title == textBoxCategorySearch.Text.Trim()

select a).FirstOrDefault();

if (listname != null)

{

listViewCategories.Items.Clear();

ListViewItem item = new ListViewItem(listname.CategoryID.ToString());

item.SubItems.Add(listname.Title);

listViewCategories.Items.Add(item);

}

else

{

MessageBox.Show("Category Name Not Found!", "Error");

}

break;

}

}

private void button1\_Click(object sender, EventArgs e)

{

DialogResult exit = MessageBox.Show("Do you really want to exit the application?", "Confirmation", MessageBoxButtons.YesNo, MessageBoxIcon.Information);

if (exit == DialogResult.Yes)

{

Application.Exit();

}

}

}

}

# Project Testing :

|  |  |  |
| --- | --- | --- |
| Operation | Discription | Output |
| Mis manager add/update/ delete user/employee information | Mis manager can add employee and based on his employee records can assign a user account for him. Each user has one employee matching record. | checked |
| Sales manger add/delete update customers | Sales manager can manipulate customer data and then update database by clicking on updata | Checked |
| Inventory controller can add/ update delete books | He will be able to manipulate book’s records and check his inventory and how much books are available | Checked |
| Oders clerks will have controlle over orders | Orders clerk can make an a order and cancel an order anytime | Checked |
| Sign in page | Each user with his userid and password can access the system and be directed to his responsible management form | Error : unable to connect order clerks with the system because my solution has two connection string and it showing me this error :  **System.Data.Entity.Core.MetadataException:** 'Schema specified is not valid. Errors:  Models.HiTechModel.ssdl(2,2) : error 0152: No Entity Framework provider found for the ADO.NET provider with invariant name 'System.Data.SqlClient'. Make sure the provider is registered in the 'entityFramework' section of the application config file. See http://go.microsoft.com/fwlink/?LinkId=260882 for more information.' |
|  |  |  |

# Conclusion:

This Project helps manage the overall performance of the systems through log analyzing. It helps you collect, correlate, and analyze the system’s data to give you an insight into the performance of your company.