**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**

**KHOA ĐÀO TẠO CHẤT LƯỢNG CAO**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

**---o0o---**



**BÁO CÁO ĐỒ ÁN 1**

**XÂY DỰNG PHẦN MỀM QUẢN LÝ**

**QUÁN ĂN TỰ CHỌN**

**SVTH 1: Đoàn Văn Long**  **17110174**

**SVTH 2: Trần Nguyên Tài 17110217**

**SVTH 3: Ngô Hoàng Minh Tâm 17110218**

**GVHD: Huỳnh Xuân Phụng**

**Tp. Hồ Chí Minh, tháng 12 năm 2019**

MỤC LỤC

[DANH MỤC CÁC HÌNH 3](#_Toc26067198)

[DANH MỤC CÁC BẢNG 3](#_Toc26067199)

[PHẦN NỘI DUNG 4](#_Toc26067200)

[I. Mô tả đồ án 4](#_Toc26067201)

[1. Lý do chọn đề tài phần mềm quản lý quán ăn tự chọn 4](#_Toc26067202)

[2. Dữ liệu, thông tin đầu vào: 4](#_Toc26067203)

[4. Giao diện dự kiến: 5](#_Toc26067204)

[II. PHÂN CÔNG CÔNG VIỆC: 6](#_Toc26067205)

[III. THIẾT KẾ: 7](#_Toc26067206)

[1. Thuật toán: 7](#_Toc26067207)

[2. Thiết kế lớp: 16](#_Toc26067208)

[3. Thiết kế cơ sở dữ liệu: 21](#_Toc26067209)

[4. Bảng mô tả các Field trong Table: 21](#_Toc26067210)

[5. Thiết kế giao diện: 24](#_Toc26067211)

[PHẦN KẾT LUẬN 29](#_Toc26067212)

[1. Ưu điểm, nhược điểm: 29](#_Toc26067213)

[a. Ưu điểm 29](#_Toc26067214)

[b. Nhược điểm 29](#_Toc26067215)

[2. Nhìn chung 29](#_Toc26067216)

[3. Bài học rút ra 29](#_Toc26067217)

[4. Hướng phát triển 30](#_Toc26067218)

[TÀI LIỆU THAM KHẢO 31](#_Toc26067219)

# DANH MỤC CÁC HÌNH

* Hình 1: Giao diện đăng nhập
* Hình 2: Giao diện Menu chính
* Hình 3: Cơ sở dữ liệu của hệ thống
* Hình 4: Nhóm các hình giao diện

*+ Hình 4.1: Giao diện chính*

*+ Hình 4.2: Giao diện đăng nhập*

*+ Hình 4.3: Giao diện thông tin Khách hàng*

*+ Hình 4.4: Giao diện thông tin Nhân viên*

*+ Hình 4.5: Giao diện Bảng chọn món*

*+ Hình 4.6: Giao diện thông tin Đăng nhập*

*+ Hình 4.7: Giao diện thông tin Món ăn*

*+ Hình 4.8: Giao diện xuất phiếu hóa đơn*

*+ Hình 4.9: Giao diện thống kê doanh thu*

# DANH MỤC CÁC BẢNG

* Bảng 1: Bảng phân công công việc và phần trăm đạt được
* Bảng 2: Bảng danh mục các lớp chính được sử dụng trong chương trình
* Bảng 3: Bảng danh mục các phương thức chính được sử dụng trong một lớp
* Bảng 4: Bảng mô tả các Table trong Cơ sở dữ liệu
* Bảng 5: Nhóm các bảng mô tả các Field trong Table
* Bảng 6: Thiết kế giao diện

# PHẦN NỘI DUNG

## I. Mô tả đồ án

### 1. Lý do chọn đề tài phần mềm quản lý quán ăn tự chọn

Trong thời đại này nay, công nghệ thông tin đang được phát triển mạnh mẽ, nhanh chóng và xâm nhập vào nhiều lĩnh vực khoa học, kỹ thuật cũng như trong cuộc sống. Nó đóng vai trò hết sức quan trọng , có thể tạo ra những bước đột phá lớn. Việc xây dựng các phần mềm quản lý để hỗ trợ cho công việc kinh doanh là một yêu cầu tất yếu. Việc quản lý một quán ăn, nhà hàng sẽ trở nên dễ dàng và trôi chảy hơn nếu có sự hỗ trợ của một phần mềm quản lý. Đó cũng chính là lý do mà nhóm chúng em quyết định thực hiện đề tài “Quản lý quán ăn tự chọn” bằng phương pháp lập trình hướng đối tượng.

Phần mềm ứng dụng công nghệ hiện đại, người quản lý có thể theo dõi mọi hoạt động kinh doanh tại quán ăn, nhà hàng vào bất cứ thời điểm nào mà không cần phải có mặt tại quán thông qua phần mềm. Bên cạnh đó, với giạo diện thân thiện với người dùng, điều này sẽ giúp đỡ cho nhiều khách hàng dù chưa dùng quen vẫn sẽ dễ tiếp cập và sử dụng phần mềm.

### 2. Dữ liệu, thông tin đầu vào:

- Xây dựng giả định một cơ sở dữ liệu gồm 4 accout Nhân Viên của hệ thống, 20 Khách hàng, 26 món ăn được phân ra các dạng thức ăn như: Thức ăn khô, thức ăn nước, đồ uống  
3. Mục đích và tính năng:

* Tiết kiệm thời gian cho khách hàng, đóng vai trò như một nhân viên bồi bàn:

+ Phần mềm quản lý quán ăn tự chọn giúp khách hàng có thể thực hiện order điện tử ngay trên máy tính bảng, ứng dụng điện thoại.

+ Thông tin gọi món ngay lập tức được chuyển tới Bếp và Thu ngân mà không cần phải di chuyển.

+ Nhờ thế, có thể hạn chế công việc cho nhân viên trong giờ cao điểm. Khách hàng không cần thiết phải chờ đợi phục vụ bàn để gọi món.

* Thông báo tự động tới quầy bếp / chế biến:

+ Tất cả yêu cầu từ Thu ngân/Bồi bàn sẽ được chuyển xuống bếp tức thì ngay khi khách hàng gọi món.

+ Hỗ trợ in hóa đơn ngay tại quầy bếp, danh sách các món ăn đều được sắp xếp theo thứ tự trước sau giúp bộ phận bếp có thể chủ động điều phối chế biến hợp lý, nhanh chóng.

+ Khi món ăn được hoàn thành, nhân viên bếp có thể thông báo tới Bồi bàn ngay lập tức.

* Phục vụ khách hàng tức thì thông qua phần mềm:

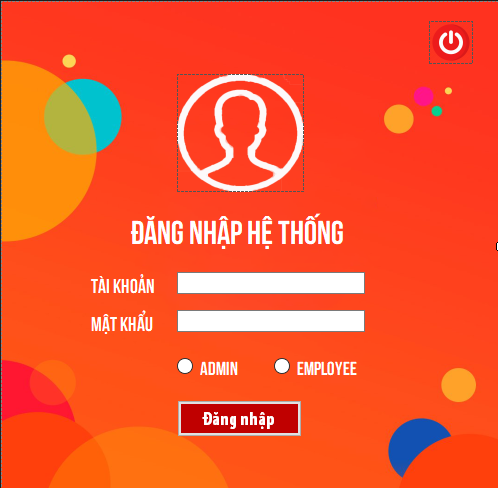
+ Với ưu điểm có thể tích hợp trên thiết bị di động, màn hình có thể hiển thị tất cả món ăn, đồ uống có tại Quán ăn.

* Thanh toán chính xác hạn chế tỷ lệ thất thoát cho thu ngân

+ Tạo hoá đơn trong tích tắc, ghi nhận tiền khách trả và tiền thừa phải trả khách, tránh sai sót trong tính toán.

+ Hệ thống lưu lại lịch sử thao tác, hóa đơn, giúp quản lý dễ dàng rà soát thông tin khi gặp vấn đề.

### 4. Giao diện dự kiến:



*Hình 1: Giao diện Đăng nhập*



*Hình 2: Giao diện Menu chính*

## II. PHÂN CÔNG CÔNG VIỆC:

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Sinh Viên** | **Khát quát công việc** | **Tỷ lệ đóng góp** |
| 1 | Đoàn Văn Long | Thiết kế cơ sở dữ liệu | 100% |
| 2 | Ngô Hoàng Minh Tâm | Bổ sung thông tin dữ liệu | 100% |
| 3 | Ngô Hoàng Minh Tâm | Thiết kế giao diện phần mềm | 100% |
| 4 | Đoàn Văn Long | Các thao tác cơ bản trong Quản  lý Nhân Viên (thêm, xóa, sửa,  cập nhật, tìm kiếm) | 100% |
| 5 | Đoàn Văn Long | Các thao tác cơ bản trong Quản  lý Khách Hàng (thêm, xóa, sửa,  cập nhật, tìm kiếm) | 100% |
| 6 | Đoàn Văn Long | Các thao tác cơ bản trong Quản  lý Đăng nhập – phân quyền  (thêm, xóa, sửa, cập nhật, tìm  kiếm) | 100% |
| 7 | Trần Nguyên Tài | Các thao tác trong việc Lựa  chọn món ăn (lựa chọn món,  xuất ra hóa đơn bán lẻ xem  trước) | 100% |
|  | Đoàn Văn Long | Các thao tác cơ bản trong Quản  lý Món Ăn (thêm, xóa, sửa, cập  nhật, tìm kiếm) | 100% |
| 8 | Trần Nguyên Tài | Xuất phiếu hóa đơn bán lẻ | 100% |
| 9 | Trần Nguyên Tài | Form Doanh Thu theo tháng | 100% |
| 10 | Ngô Hoàng Minh Tâm | Viết báo cáo (doc) | 100% |
| 11 | Ngô Hoàng Minh Tâm | Làm file trình chiếu (ppt) | 100% |
| 12 | Ngô Hoàng Minh Tâm | Form đăng nhập hệ thống | 100% |

*Bảng 1: Bảng phân công công việc và phần trăm đạt được*

## III. THIẾT KẾ:

### Thuật toán:

#### Các phương thức ở Tầng dữ liệu

##### Lớp DataContext

public DbSet<Customer> Customers { get; set; }

public DbSet<Employee> Employees { get; set; }

public DbSet<Food> Foods { get; set; }

public DbSet<Category> Categories { get; set; }

public DbSet<Bill> Bills { get; set; }

public DbSet<Detail> Details { get; set; }

public DbSet<Account> Accounts { get; set; }

##### Lớp Account: Tạo đối tượng Đăng nhập

[Table("Account")]

public class Account

{

[Key]

[DatabaseGenerated(DatabaseGeneratedOption.None)]

[Required]

[MaxLength(100)]

public string Username { get; set; }

[Required]

[MinLength(8)]

public string Password { get; set; }

[Required]

public bool IsAdmin { get; set; }

[DisplayName("Họ tên")]

[Required]

[MaxLength(50)]

public string Name { get; set; }

public DateTime DOB { get; set; }

public string Phone { get; set; }

public string Sex { get; set; }

}

##### Lớp Bill: Tạo đối tượng Hóa Đơn

[Table("Bill")]

public class Bill

{

[Key]

public int ID { get; set; }

public DateTime BillDate { get; set; }

public virtual Customer Customer { get; set; }

public virtual Employee Employee { get; set; }

}

##### Lớp Category: Tạo đối tượng Loại Món Ăn

[Table("Category")]

public class Category

{

public int ID { get; set; }

[Required]

[MaxLength(50)]

public string Name { get; set; }

}

##### Lớp Person: Tạo đối tượng Người

public class Person

{

[Key]

public int ID { get; set; }

[DisplayName("Họ tên")]

[Required]

[MaxLength(50)]

public string Name { get; set; }

[DisplayName("Địa chỉ")]

[Required]

public string Address { get; set; }

[DisplayName("Ngày sinh")]

[Required]

public DateTime DOB { get; set; }

[DisplayName("SĐT")]

[Required]

[MaxLength(25)]

public string Phone { get; set; }

[DisplayName("Giới tính")]

[Required]

[MaxLength(3)]

public string Sex { get; set; }

[DisplayName("Hình ảnh")]

public string Image { get; set; }

}

##### Lớp Employee: tạo đối tượng cho Nhân Viên (áp dựng Kế Thừa lớp Person)

[Table("Employee")]

public class Employee : Person

{

public DateTime HireDate { get; set; }

}

##### Lớp Customer: tạo đối tượng cho Khách Hàng (áp dựng Kế Thừa lớp Person)

[Table("Customer")]

public class Customer : Person

{

public int Wallet { get; set; }

}

##### Lớp Detail: Tạo đối tượng Chi tiết hóa đơn

[Table("Detail")]

public class Detail

{

[Key]

public int ID { get; set; }

public int Amount { get; set; }

public int Bill\_ID { get; set; }

[ForeignKey("Bill\_ID")]

public virtual Bill Bill { get; set; }

public int Food\_ID { get; set; }

[ForeignKey("Food\_ID")]

public virtual Food Food { get; set; }

}

##### Lớp Food: Tạo đối tượng cho Món ăn

[Table("Food")]

public class Food

{

[Key]

public int ID { get; set; }

[DisplayName("Họ tên")]

[Required]

[MaxLength(25)]

public string Name { get; set; }

[DisplayName("Đơn giá")]

[Required]

public float Price { get; set; }

[DisplayName("ĐVT")]

[Required]

[MaxLength(25)]

public string Unit { get; set; }

public int Category\_ID { get; set; }

[ForeignKey("Category\_ID")]

public virtual Category Category { get; set; }

#### Các phương thức ở Tầng nghiệp vụ

##### Lớp BL Menu:

+ Hàm GetFoodByCategory(): hàm lấy thông tin các món ăn dựa trên Mã loại món ăn (CategoryID)

public static List<dynamic> GetFoodByCategory(int CategoryID)

{

using (var ctx = new StoreContext())

{

var list = (from t in ctx.Foods

join p in ctx.Categories on t.Category\_ID equals p.ID

where t.Category\_ID == CategoryID

select new

{

t.ID ,

t.Name,

t.Price,

t.Unit,

CategoryName=p.Name

}).Distinct();

return list.ToList<dynamic>();

}

}

+ Hàm GetListCategory(): hàm lấy toàn bộ thông tin của Các loại món ăn

public static List<Category> GetListCategory()

{

using (var ctx = new StoreContext())

{

return ctx.Categories.ToList();

}

}

##### Lớp BL Bill:

+ Hàm AddBill(): thêm vào thông tin hóa đơn. Lấy ID của nhân viên đang đăng nhập hệ thống để Tạo hóa đơn, nhân viên này sẽ chịu trách nhiệm về hóa đơn đã tạo.

public static int AddBill(int EmployeeID)

{

using (var ctx = new StoreContext())

{

var bill = new Bill()

{

BillDate = DateTime.Now,

Employee\_ID = EmployeeID,

Total = 0,

};

ctx.Bills.Add(bill);

ctx.SaveChanges();

return bill.ID;

}

}

+ Hàm ThanhToan(): Hàm này sẽ truy xuất về thông tin của Người mua hàng (Customer). Nếu hành động thanh toán xảy ra, Ví tiền trong tài khoản người mua hàng sẽ bị trừ vào số tiền thanh toán đơn hàng.

public static void ThanhToan(int ToTal,int Customer\_ID)

{

using (var ctx = new StoreContext())

{

var dt = ctx.Customers.Where(x => x.ID == Customer\_ID).FirstOrDefault();

if (dt != null)

{

dt.Wallet = dt.Wallet-ToTal;

ctx.SaveChanges();

}

}

}

+ Hàm AddCustomerToBill(): Hàm này sẽ kiểm tra trong danh sách khách hàng, những thông tin khách hàng để lưu thông tin vào hóa đơn.

public static void AddCustomerToBill(int CustomerID, int Bill\_ID)

{

using (var ctx = new StoreContext())

{

var dt = ctx.Bills.Where(x => x.ID == Bill\_ID).FirstOrDefault();

if (dt != null)

{

dt.Customer\_ID = CustomerID;

ctx.SaveChanges();

}

}

}

+ Hàm GetWallet(): Lấy thông tin ví tiền, củ thể là trả về số tiền hiện tại trong Ví thanh toán của Khách hàng

public static int GetWallet(int CustomerID)

{

using (var ctx = new StoreContext())

{

var dt = ctx.Customers.Where(x => x.ID == CustomerID).FirstOrDefault();

return dt.Wallet;

}

}

+ Hàm LietKe(): liệt kê những thông tin cơ bản của hóa đơn như: Số hóa đơn, ngày xuất hóa đơn, tên nhân viên phụ trách và tổng hóa đơn, qua việc kết hợp 2 bảng Employee và Bill để xuất ra tên của nhân viên phụ trách hóa đơn.

public static List<dynamic> LietKe(DateTime start,DateTime end)

{

using (var ctx = new StoreContext())

{

var list = (from b in ctx.Bills

join nv in ctx.Employees on b.Employee\_ID equals nv.ID

where b.BillDate >= start && b.BillDate <= end

select new

{

b.ID,

b.BillDate,

nv.Name,

b.Total

}).Distinct();

return list.ToList<dynamic>();

}

}

+ Hàm ThongKe(): hàm dùng kể thống kê tổng số tiền doanh thu của quán ăn trong một khoảng thời gian nhất định.

public static int? ThongKe(DateTime start,DateTime end)

{

using (var ctx = new StoreContext())

{

var total = ctx.Bills.Where(t => t.BillDate >= start ).Where(t => t.BillDate <= end).Sum(t => t.Total);

return total;

}

}

+ Hàm GetBill(): hàm lấy thông tin hóa đơn

public static List<dynamic> GetBill(int ID\_Bill)

{

using (var ctx = new StoreContext())

{

var list = (from dt in ctx.Details.AsEnumerable()

join b in ctx.Bills on dt.Bill\_ID equals b.ID

join f in ctx.Foods.AsEnumerable() on dt.Food\_ID equals f.ID

where b.ID == ID\_Bill

select new

{

f.Name,

f.Unit,

f.Price,

dt.Amount,

Total = Convert.ToInt32(f.Price) \* Convert.ToInt32(dt.Amount)

}).Distinct();

return list.ToList<dynamic>();

}

}

##### Lớp BL Detail

+ Hàm AddOrUpdateDetail(): hàm thêm hay cập nhật thông tin cho hóa đơn.

public static int AddOrUpdateDetail(int IDFood,int IDBill,int SoLuong)

{

using (var ctx = new StoreContext())

{

var dt = ctx.Details.Where(x => x.Food\_ID == IDFood).Where(x=>x.Bill\_ID==IDBill).FirstOrDefault();

if (dt != null)

{

dt.Amount = SoLuong;

ctx.SaveChanges();

return dt.ID;

}

else

{

var dtAdd = new Detail()

{

Food\_ID = IDFood,

Bill\_ID = IDBill,

Amount = SoLuong

};

ctx.Details.Add(dtAdd);

ctx.SaveChanges();

return dtAdd.ID;

}

}

}

+ Hàm DeleteDetail(): hàm xóa thông tin cho hóa đơn.

public static void DeleteDetail(int ID\_Food,int ID\_Bill)

{

using (var ctx = new StoreContext())

{

var dt = ctx.Details.Where(x => x.Food\_ID == ID\_Food).Where(x => x.Bill\_ID == ID\_Bill).FirstOrDefault();

ctx.Details.Remove(dt);

ctx.SaveChanges();

}

}

#### Các phương thức ở Tầng giao diện

##### Lớp FormFoodChoices: Lựa chọn Món ăn và xem Giỏ hàng đã chọn

+ Hàm chọn món ăn: Khách hàng chọn món cần sự dụng NumericUpDown để chọn số lượng món trước khi nhấn chọn món. Khi nhấn chọn món, nền khung của món ăn sẽ chuyển sang màu vàng, ghi nhận món ăn này đã được chọn. Theo đó, món ăn sẽ được lưu trong datagridview của hóa đơn bên cạnh bảng chọn món. Hóa đơn sẽ chỉ ra Tên (Name), đơn vị (Unit), số lượng (Amount), và giá (Price) sử dụng hàm GetBill trên lớp BLBill. Thêm vào cột Total để tính tổng giá của từng món ăn được chọn. Cuối cùng xuất ra textbox2 là tổng hóa đơn mà khách hàng phải trả.

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

{

Button b = sender as Button;

var pr = b.Parent;

var num = pr.Controls.Find("SL", false).First() as NumericUpDown;

if (num.Value > 0)

{

BLDetails.AddOrUpdateDetail(Convert.ToInt32(b.Tag.ToString()), ID\_Bill, Convert.ToInt32(num.Value));

pr.BackColor = Color.Yellow;

dataGridView1.DataSource = BLBill.GetBill(ID\_Bill);

dataGridView1.Columns["Price"].DefaultCellStyle.Format = "#,##0";

dataGridView1.Columns["Total"].DefaultCellStyle.Format = "#,##0";

textBox2.Text=string.Format("{0:#,##0}", BLBill.TotalByBill(ID\_Bill));

}

}

+ Hàm Bỏ chọn món: Tương tự như hàm chọn món, hàm này sẽ giúp khách hàng loại bỏ những món ăn không mong muốn đã chọn

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

{

…

BLDetails.DeleteDetail(Convert.ToInt32(b.Tag.ToString()), ID\_Bill);

…

}

##### Lớp FormEmployee: Quản lý thông tin nhân viên bằng các phương thức thêm, xóa, sửa, tìm kiếm, cập nhật

+ Hàm Cập nhật Nhân viên: Khi admin có sửa đổi hay bổ sung thông tin của bất kỳ nhân viên nào, admin sẽ bấm chọn nút Cập nhật, thông tin sẽ được làm mới. Nếu Admin vừa Thêm vào một nhân viên mới thực hiện new Employee() rồi kết quả “Đã thêm xong”, ngược lại Admin thực hiện sửa thông tin nhân viên thì ghi nhận kết quả “Đã sửa xong”

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

{

if (Them)

{

using (StoreContext db = new StoreContext())

{

var employee = new Employee();

employee.Name = txtHoTen.Text;

employee.DOB = dtpDOB.Value.Date;

employee.Address = txtDiaChi.Text;

employee.Phone = txtSoDT.Text;

employee.Sex = (rdbNam.Checked) ? "Nam" : "Nữ";

employee.HireDate = dtpHireDate.Value.Date;

employee.Image = imagepath;

db.Employees.Add(employee);

db.SaveChanges();

}

LoadData();

resettext();

//// Không cho thao tác trên các nút Lưu / Hủy

btnCapNhat.Enabled = false;

btnHuy.Enabled = false;

//// Cho thao tác trên các nút Thêm / Sửa / Xóa / Thoát

btnThem.Enabled = true;

btnEdit.Enabled = true;

btnDel.Enabled = true;

// Thông báo

MessageBox.Show("Đã thêm xong!");

}

else

{

try

{

// Thứ tự dòng hiện hành

int r = dgvNV.CurrentCell.RowIndex;

// MaKH hiện hành

int idEm = Convert.ToInt32(dgvNV.Rows[r].Cells[0].Value.ToString());

// Câu lệnh SQL

StoreContext db = new StoreContext();

var khQuery = (from em in db.Employees

where em.ID == idEm

select em).SingleOrDefault();

if (khQuery != null)

{

khQuery.Name = txtHoTen.Text;

khQuery.DOB = dtpDOB.Value.Date;

khQuery.Address = txtDiaChi.Text;

khQuery.Phone = txtSoDT.Text;

khQuery.Sex = (rdbNam.Checked) ? "Nam" : "Nữ";

khQuery.HireDate = dtpHireDate.Value.Date;

khQuery.Image = imagepath;

db.SaveChanges();

LoadData();

}

// Load lại dữ liệu trên DataGridView

LoadData();

resettext();

groupBox1.Enabled = false;

picEmployee.Image = null;

//// Không cho thao tác trên các nút Lưu / Hủy

btnCapNhat.Enabled = false;

btnHuy.Enabled = false;

//// Cho thao tác trên các nút Thêm / Sửa / Xóa / Thoát

btnThem.Enabled = true;

btnEdit.Enabled = true;

btnDel.Enabled = true;

// Thông báo

MessageBox.Show("Đã sửa xong!");

}

catch

{

MessageBox.Show("Không sửa được. Lỗi rồi!");

}

}

+ Hàm Xóa Nhân viên: Khi admin chọn xóa thông tin nhân viên, hệ thống muốn chắc chắn việc này sẽ hiện viên cảnh báo “Bạn có chắc xóa mẫu tin này không?” Xóa khi đã chắc chắn. Thực hiện RemoveRange() để xóa thông tin nhân viên đã chọn, xuất ra “Xóa thành công” để báo cáo cho Admin. Try … catch … sẽ báo lỗi việc xóa không thành công khi phát sinh lỗi dữ liệu.

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

{

try

{

// Hiện thông báo xác nhận việc xóa mẫu tin

// Khai báo biến traloi

DialogResult traloi;

// Hiện hộp thoại hỏi đáp

traloi = MessageBox.Show("Chắc xóa mẫu tin này không?", "Trả lời", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

// Kiểm tra có nhắp chọn nút Ok không?

if (traloi == DialogResult.Yes)

{

try

{

StoreContext db = new StoreContext();

int idEm = Convert.ToInt32(txtMaNV.Text);

var nvQuery = from em in db.Employees

where em.ID == idEm

select em;

db.Employees.RemoveRange(nvQuery);

db.SaveChanges();

// Cập nhật lại DataGridView

LoadData();

// Thông báo

MessageBox.Show("Đã xóa xong!");

}

catch

{

MessageBox.Show("Không xóa được. Lỗi rồi!!!");

}

}

}

catch (SqlException)

{

MessageBox.Show("Không xóa được. Lỗi rồi");

}

}

+ Hàm lấy ảnh: Ở form này có thêm hàm lấy ảnh để lấy hình ảnh nhân viên từ một đường link trên thư mục ảnh. Hình ảnh sẽ được lưu lại trong cơ sở dữ liệu ở dạng một đường link.

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

{

using (OpenFileDialog ofd = new OpenFileDialog() { Filter = "JPEG|\*.jpg" })

{

if (ofd.ShowDialog() == DialogResult.OK)

{

picEmployee.Image = System.Drawing.Image.FromFile(ofd.FileName);

imagepath = ofd.FileName;

}

}

}

##### Lớp FormCustomer và FormFoodManager, FormLoginManage: gồm các phương thức thêm, xóa, sửa, tìm kiếm, cập nhập tương tự như FormEmployee

##### Lớp FormLogin: Đăng nhập vào hệ thống

+ Hàm đăng nhập: Hàm này sẽ cho phép người dùng đăng nhập vào tài khoản. Người dùng ở đây có 2 phần quyền: Admin và Employee. Khi đăng nhập ở phân quyền Admin, người dùng sẽ được chuyển thẳng đến form Quản lý, ngược lại khi đăng nhập là Employee, người dùng sẽ được chuyển đến Form chọn món để chọn món ăn cho khách hàng.

private void button1\_Click\_1(object sender, EventArgs e)

{

if (IsUser(txtUsername.Text,txtPassword.Text, rbAdmin.Checked))

{

if (rbAdmin.Checked)

{

FormManage f1 = new FormManage();

this.Hide();

f1.ShowDialog();

}

if (rbEmployee.Checked)

{

FormFoodChoices f1 = new FormFoodChoices(EmployeeID);

this.Hide();

f1.ShowDialog();

}

}

+ Hàm kiểm tra User: lấy thông tin của User vừa nhập đối chiếu thông tin đã lưu trong cơ sở dữ liệu

private bool IsUser(string username, string password, bool quyen)

{

StoreContext db = new StoreContext();

var q = (from p in db.Accounts

where p.Username == username

&& p.Password == password && p.IsAdmin==quyen

select p).SingleOrDefault();

if (q != null)

{

EmployeeID = q.Employee\_ID;

return true;

}

else return false;

}

### Thiết kế lớp:

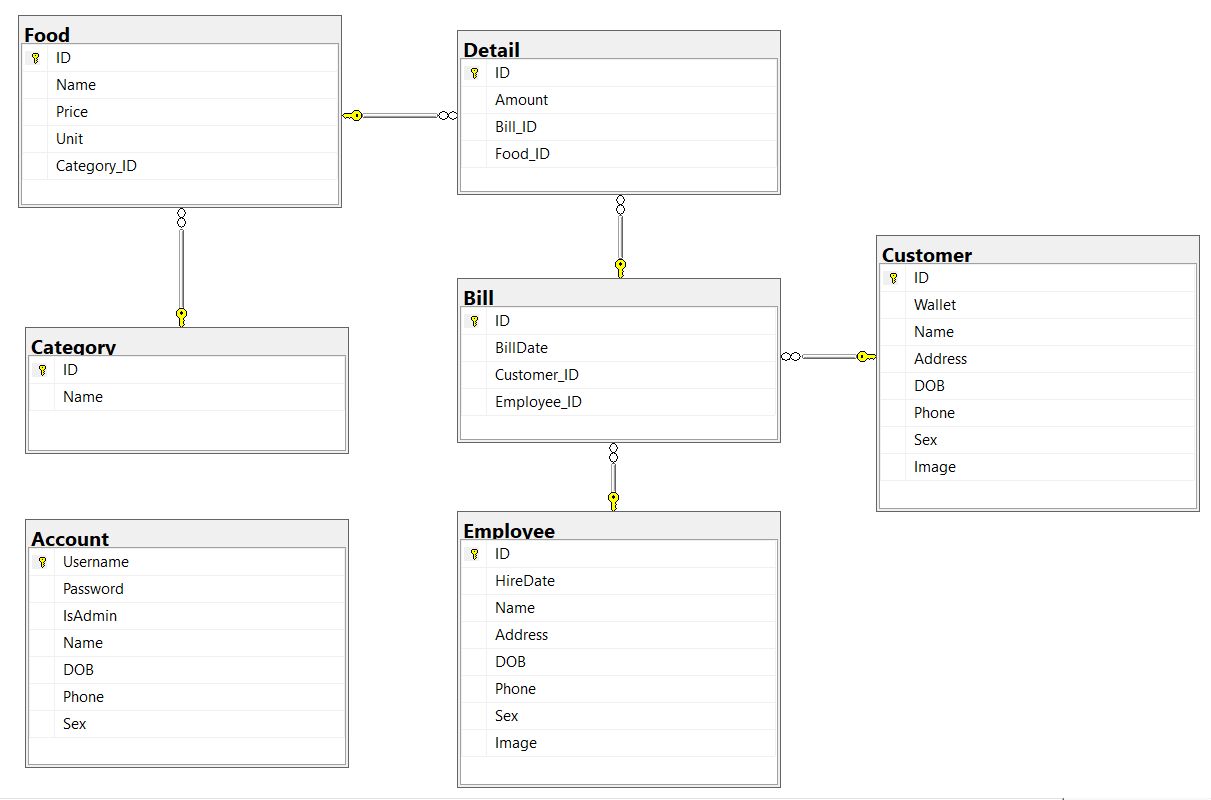
|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên lớp** | **Mục đích chính của lớp trong chương trình** | **SV phụ trách** |
| 1 | FormLogin | Lớp hiển thị giao diện xử lý Đăng nhập | Ngô Hoàng Minh Tâm |
| 2 | Form  FoodChoices | Lớp hiển thị giao diện Chọn món ăn và xuất ra danh mục những món ăn đã chọn, tính toán tổng hóa đơn. | Trần Nguyên Tài |
| 3 | Form  Employee | Lớp hiển thị giao diện Quản lý Nhân viên, gồm tính năng tìm kiếm, thêm, xóa, sửa, cập nhật thông tin. (được liên kết từ Form Manage) | Đoàn Văn Long |
| 4 | FormCustomer | Lớp hiển thị giao diện Quản lý Khách hàng, gồm tính năng tìm kiếm, thêm, xóa, sửa, cập nhật thông tin. (được liên kết từ Form Manage) | Đoàn Văn Long |
| 5 | Form  LoginManage | Lớp hiển thị và xử lí quyền trong bảng DangNhap (được liên kết từ Form Manage) | Đoàn Văn Long |
| 6 | FormManage | Lớp hiển thị giao diện Quản lý dành cho Admin, gồm các menu Quản lý nhân viên, khách hàng, Quản lý doanh thu, Quản lý đăng nhập, món ăn  (chỉ admin mới có thể truy cập) | Ngô Hoàng Minh Tâm |
| 7 | ThongKeDoanhThu | Lớp Thống kê doanh thu của quán ăn (được liên kết từ Form Manage) | Trần Nguyên Tài |
| 8 | ReportBill | Lớp xuất hóa đơn bán lẻ, lưu thông tin hóa đơn tính tổng hóa đơn | Trần Nguyên Tài |
| 9 | FoodManager | Lớp hiển thị giao diện Quản lý Món ăn, gồm tính năng tìm kiếm, thêm, xóa, sửa, cập nhật thông tin. (được liên kết từ Form Manage) | Đoàn Văn Long |
| 10 | BLBill | Tầng nghiệp vụ của bảng Bill chứa các truy vấn đến Database, cùng với các hàm để ghi nhận việc thanh toán hóa đơn, thiết lập thông tin hóa đơn | Trần Nguyên Tài |
| 11 | BLDetail | Tầng nghiệp vụ của bảng Detail chứa các truy vấn đến Database. | Trần Nguyên Tài |
| 12 | BLMenu | Tầng nghiệp vụ của bảng Food chứa các truy vấn đến Database, thiết lập giao diện tự động khi chạy đến FormFoodChoices. | Trần Nguyên Tài |

*Bảng 2: Danh mục các lớp chính được sử dụng trong chương trình*

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Phương thức** | **Mục đích** | **Tên file** |
| 1 | **GetFoodByCategory(int CategoryID)**  Intput: CategoryID  Output: List<dynamic>  Pseudo code:  from t in ctx.Foods join p in ctx.Categories  on t.Category\_ID equals p.ID  where t.Category\_ID == CategoryID  select new {t.ID , t.Name, t.Price, t.Unit, CategoryName=p.Name}  Câu lệnh SQL liên kết 2 bảng Food và Category với nhau để lấy được thông tin Tên Loại Món ăn thông qua thuộc tính Mã Loại Món ăn | Hàm lấy thông tin các món ăn dựa trên Mã loại món ăn (CategoryID) | BL Menu.cs  (12) |
| 2 | **GetListCategory():**  Output: List<dynamic> | Hàm lấy toàn bộ thông tin của Các loại món ăn | BL Menu.cs  (31) |
| 3 | **AddBill(int EmployeeID)**  Input: EmployeeID  Output: List<dynamic> | Thêm vào thông tin hóa đơn. Lấy ID của nhân viên đang đăng nhập hệ thống để Tạo hóa đơn, nhân viên này sẽ chịu trách nhiệm về hóa đơn đã tạo. | BL Bill.cs  (16) |
| 4 | **ThanhToan(intToTal,intCustomer\_ID)**  Input: ToTal, Customer\_ID  Output: List<dynamic>  Pseudo code:  var dt = ctx.Customers.Where(x => x.ID = Customer\_ID).FirstOrDefault();  if (dt != null)  {  dt.Wallet = dt.Wallet-ToTal;  ctx.SaveChanges();  }  Nếu hành động thanh toán xảy ra, Ví tiền trong tài khoản người mua hàng (Wallet) sẽ bị trừ vào số tiền thanh toán đơn hàng (ToTal). | Hàm này sẽ truy xuất về thông tin của Người mua hàng (Customer). | BL Bill.cs  (31) |
| 5 | **AddCustomerToBill(int CustomerID, int Bill\_ID)**  Input: ToTal, Customer\_ID  Output: List<dynamic> | Hàm này sẽ kiểm tra trong danh sách khách hàng, những thông tin khách hàng để lưu thông tin vào hóa đơn. | BL Bill.cs  (43) |
| 6 | **GetWallet(int CustomerID)**  Input: Customer\_ID  Output: List<dynamic> | Lấy thông tin ví tiền, củ thể là trả về số tiền hiện tại trong Ví thanh toán của Khách hàng | BL Bill.cs  (64) |
| 7 | **LietKe(DateTime start, DateTime end)**  Input: start, end  Output: List<dynamic>  Pseudo code:  from b in ctx.Bills  join nv in ctx.Employees on b.Employee\_ID equals nv.ID  where b.BillDate >= start &&  b.BillDate <= end  select new{b.ID, b.BillDate, nv.Name, b.Total}  Câu lệnh SQL liên kết 2 bảng Bill và Employee với nhau để lấy được thông tin Tên Nhân Viên phụ trách hóa đơn. | Liệt kê những thông tin cơ bản của hóa đơn như: Số hóa đơn, ngày xuất hóa đơn, tên nhân viên phụ trách và tổng hóa đơn | BL Bill.cs  (72) |
| 8 | **ThongKe(DateTime start,DateTime end)**  Input: start, end  Output: List<dynamic>  Pseudo code:  var total = ctx.Bills.  Where(t => t.BillDate >= start ).  Where(t => t.BillDate <= end).  Sum(t => t.Total);  return total;  Câu lệnh giới hạn khoảng thời gian đầu vào, tính toán tổng cộng doanh thu từ tổng của từng hóa đơn | Hàm dùng kể thống kê tổng số tiền doanh thu của quán ăn trong một khoảng thời gian nhất định. | BL Bill.cs  (89) |
| 9 | **GetBill(int ID\_Bill)**  Input: ID\_Bill  Output: List<dynamic> | Hàm lấy thông tin hóa đơn | BL Bill.cs  (111) |
| 10 | **AddOrUpdateDetail (int IDFood, int IDBill, int SoLuong)**  Input: IDFood, IDBill, SoLuong  Output: List<dynamic> | Hàm thêm hay cập nhật thông tin cho hóa đơn. | BL Detail.cs  (12) |
| 11 | **DeleteDetail(int ID\_Food,int ID\_Bill)**  Input: IDFood, IDBill  Output: List<dynamic> | Hàm xóa thông tin cho hóa đơn | BL Detail.cs  (38) |
| 12 | **IsUser (string username, string password, bool quyen)**  Input: username, password, quyen  Output: true, false | Lấy thông tin của User vừa nhập đối chiếu thông tin đã lưu trong cơ sở dữ liệu | Form  Login.cs  (40) |

*Bảng 3: Danh mục các phương thức chính trong một lớp*

### Thiết kế cơ sở dữ liệu:



*Hình 3: Cơ sở dữ liệu của hệ thống*

|  |  |  |
| --- | --- | --- |
| **STT** | **Tên bảng** | **Mục đích** |
| 1 | ACCOUNT | Thuộc tính quản lý đăng nhập |
| 2 | BILL | Thuộc tính quản lý Hóa Đơn |
| 3 | CATEGORY | Thuộc tính quản lý Loại Món Ăn (Nước, Khô,  Đồ Uống) |
| 4 | CUSTOEMR | Thuộc tính quản lý Khách Hàng |
| 5 | DETAIL | Thuộc tính quản lý chi tiết Hóa Đơn |
| 6 | EMPLOYEE | Thuộc tính quản lý Nhân Viên |
| 7 | FOOD | Thuộc tính quản lý Món Ăn |

*Bảng 4: Bảng mô tả các Table trong Cơ sở dữ liệu*

### Bảng mô tả các Field trong Table:

##### Table Account: (người dùng ở đây có nghĩa là Nhân Viên)

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên trường** | **Kiểu dữ liệu** | **Mục đích** |
| 1 | Username | string[MaxLen(100)] | Tên đăng nhập người dùng |
| 2 | Password | string[MinLen(8)] | Mật khẩu người dùng |
| 3 | IsAdmin | bool | Kiểm tra xem có phải Admin không |
| 4 | Name | string[MaxLen(50)] | Tên người dùng |
| 5 | DOB | datetime | Ngày sinh người dùng |
| 6 | Phone | string | Số điện thoại người dùng |
| 7 | Sex | string | Giới tính người dùng |

##### Table Bill:

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên trường** | **Kiểu dữ liệu** | **Mục đích** |
| 1 | Bill\_ID | int | Mã Hóa đơn |
| 2 | BillDate | Datetime | Ngày lập hóa đơn |
| 3 | Customer\_ID | int | Mã khách hàng |
| 4 | Employee\_ID | Int | Mã nhân viên |

##### Table Category:

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên trường** | **Kiểu dữ liệu** | **Mục đích** |
| 1 | Category\_ID | int | Mã Loại Món Ăn |
| 2 | Name | string | Tên Loại Món ăn |

##### Table Customer:

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên trường** | **Kiểu dữ liệu** | **Mục đích** |
| 1 | Customer\_ID | int | Mã Khách hàng |
| 2 | Name | string[MaxLen(50)] | Tên Khách hàng |
| 3 | Address | String | Địa chỉ của Khách hàng |
| 4 | DOB | datetime | Ngày sinh Khách hàng |
| 5 | Phone | String | Số điện thoại Khách hàng |
| 6 | Sex | String | Giới tính Khách hàng |
| 7 | Image | String | Ảnh Khách hàng |
| 8 | Wallet | int | Ví tiền trong tài khoản của KH |

##### Table Employee:

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên trường** | **Kiểu dữ liệu** | **Mục đích** |
| 1 | Employee\_ID | int | Mã Nhân viên |
| 2 | Name | string[MaxLen(50)] | Tên Nhân viên |
| 3 | Address | String | Địa chỉ của Nhân viên |
| 4 | DOB | datetime | Ngày sinh Nhân viên |
| 5 | Phone | String | Số điện thoại Nhân viên |
| 6 | Sex | String | Giới tính Nhân viên |
| 7 | Image | String | Ảnh Nhân viên |
| 8 | HireDate | datetime | Ngày vào làm của Nhân viên |

##### Table Food:

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên trường** | **Kiểu dữ liệu** | **Mục đích** |
| 1 | Food\_ID | int | Mã Nhân viên |
| 2 | Name | string[MaxLen(25)] | Tên Nhân viên |
| 3 | Price | Float | Địa chỉ của Nhân viên |
| 4 | Unit | String | Ngày sinh Nhân viên |
| 5 | Category\_ID | int | Số điện thoại Nhân viên |

##### Table Detail:

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên trường** | **Kiểu dữ liệu** | **Mục đích** |
| 1 | Datail\_ID | int | Mã Chi tiết hóa đơn |
| 2 | Amount | Int | Số lượng |
| 3 | Bill\_ID | Int | Mã Hóa đơn |
| 4 | Food\_ID | int | Mã Món Ăn |

### Thiết kế giao diện:

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Màn hình/Cửa sổ/Dialog** | **Giải thích ngắn gọn** | **Mục đích chính của màn hình** |
| 1 | Màn hình giao diện chính:    *Hình 4.1: Giao diện chính* | Hiện ra những button với các chức năng tương ứng. (nhóm quyền Admin được phép truy cập) | Hiển thị ra giao diện chính của phần mềm. |
| 2 | Màn hình đăng nhập :    *Hình 4.2: Giao diện đăng nhập* | Lấy thông tin và quyền người dùng tương ứng khi đăng nhập vào. | Hiển thị giao diện đăng nhập vào phần mềm. |
| 3 | Màn hình bảng khách hàng :    *Hình 4.3: Giao diện thông tin Khách hàng* | Cho phép thêm hoặc chỉnh sửa, xóa nhân viên (nhóm quyền Admin, Staff được phép truy cập). | Hiển thị giao diện xem thông tin cá nhân của Khách Hàng |
| 4 | Màn hình bảng nhân viên:    *Hình 4.4: Giao diện thông tin Nhân viên* | Cho phép thêm hoặc chỉnh sửa, xóa nhân viên (nhóm quyền Admin được phép truy cập). | Hiển thị giao diện nhân viên. |
| 5 | Màn hình bảng chọn món:    *Hình 4.5: Giao diện Bảng chọn món* | Cho phép khách hàng chọn Món với các Tab: Món ăn Khô, món ăn Nước, Nước giải khát | Hiển thị giao diện bảng chọn món |
| 6 | Màn hình bảng chọn món:    *Hình 4.6: Giao diện thông tin Đăng nhập* | Cho phép thêm hoặc chỉnh sửa, xóa thông tin đăng nhập của các nhân viên (nhóm quyền Admin được phép truy cập). | Hiện thị giao diện quản lý đăng nhập |
| 7 | Màn hình bảng món ăn:    *Hình 4.7: Giao diện thông tin Món ăn* | Cho phép thêm hoặc chỉnh sửa, xóa món ăn (nhóm quyền Admin được phép truy cập). | Hiển thị giao diện quản lý món ăn |
| 8 | Màn hình Phiếu hóa đơn    *Hình 4.8: Giao diện xuất phiếu hóa đơn* | Xuất ra Phiếu hóa đơn để khách hàng xem lại Thông tin hóa đơn vừa thanh toán. Trong phiếu thanh toán có bao gồm: tên nhân viên xuất phiếu, tên khách hàng, ngày xuất phiếu, tổng giá trị đơn mua | Hiển thị giao diện Report Hóa đơn |
| 9 | Màn hình Thống kê doanh thu:    *Hình 4.9: Giao diện thống kê doanh thu* | Cho phép Amin xem lại và thống kê tổng doanh thu trong từng khoảng thời gian | Hiển thị giao diện thống kê doanh thu |

*Bảng 6: Thiết kế giao diện*

# PHẦN KẾT LUẬN

## 1. Ưu điểm, nhược điểm:

### a. Ưu điểm

- Giao diện đẹp, ưa nhìn, đơn giản. Giúp cho người dùng mới dễ tiếp cận và sử dụng

- Đã ứng dụng được kiến thức Kế thừa vào thiết kế Database (lớp Employee và lớp Customer là lớp con của lớp Peson)

### b. Nhược điểm

- Vì chưa chuyển hình ảnh sang dạng tối ưu nên khi kéo scroll sẽ bị chậm đối với 1 số máy tính

## 2. Nhìn chung

Trong quá trình thực hiện đề tài, mặc dù đã ra sức nghiên cứu và cố gắng nhưng chắc chắn khó tránh khỏi những thiếu sót, chúng em rất mong nhận được sự chỉ dẫn, ý kiến đóng góp của thầy để chúng em ngày càng hoàn thiện hơn.

Sau một thời học tập và nghiên cứu, em nhận thấy một số khó khăn:

+ Chúng em chưa rõ cách áp dụng triệt để Kế thừa vào bài quản lý

+ Việc tham khảo và viết ra báo cáo đồ án hoàn chỉnh cũng là một khó khăn lớn. Cụ thể là cách điễn đạt cho các đoạn code được sử dụng trong đồ án cũng như trình tự các phần mục trong đồ án.

+ Bên cạnh đó, chúng em cũng gặp phải khó khăn trong việc thực hiện theo kế hoạch. Cụ thể là có một số thay đổi về người thực hiện công việc và một số trì hoãn về thời gian hoàn thành so với thời gian dự kiến.

## 3. Bài học rút ra

Sau quá trình thực hiện đề tài, chúng em đã đúc kết thêm được một số kinh nghiệm có thể giúp cho các đồ án tiếp theo có thể tốt hơn. Cụ thể là:

+ Cần phải chủ động hơn về thời gian, dự trù trước thời hạn hoàn thành để đi đúng theo thời gian hoàn thành dự kiến đã đề ra.

+ Cần phải dành nhiều thời gian hơn trong việc tìm hiểu về đồ án để có thể hiểu rõ mình đang làm gì.

+ Cần phải họp nhóm nhiều hơn là mỗi người tự làm. Vì đôi khi ban đầu chia theo kế hoạch, nhưng người này có khả năng làm được công việc của người kia, và ngược lại. Điều này còn giúp cho cả 3 hiểu được mình đang làm gì và sẽ làm gì cùng lúc, đỡ phải gặp trường hợp 3 người không hiểu ý nhau, hoặc 1 người làm xong code nhưng không phù hợp với người kia, tránh làm chậm tiến trình công việc.

## 4. Hướng phát triển

Vì không có đủ kiến thức, cũng như thời gian nên chúng em chưa thể làm tốt nhất trong đồ án lần này. Vì vậy, nếu có thêm nhiều cơ hội chúng em tự hứa phát triển đồ án này tốt hơn bây giờ. Sau đây là một số hướng đi khác để phát triển đồ án trở nên tốt hơn:

+ Tìm hiểu thêm các cách ứng dụng Kế thừa để có thể áp dụng triệt để hơn khi Kế thừa các phương thức Thêm, Xóa, Sửa, Tìm kiếm, Cập Nhật

+ Thiết kế phần mềm tương thích với nhiều công cụ: điện thoại di động, máy tính bảng,…

# TÀI LIỆU THAM KHẢO