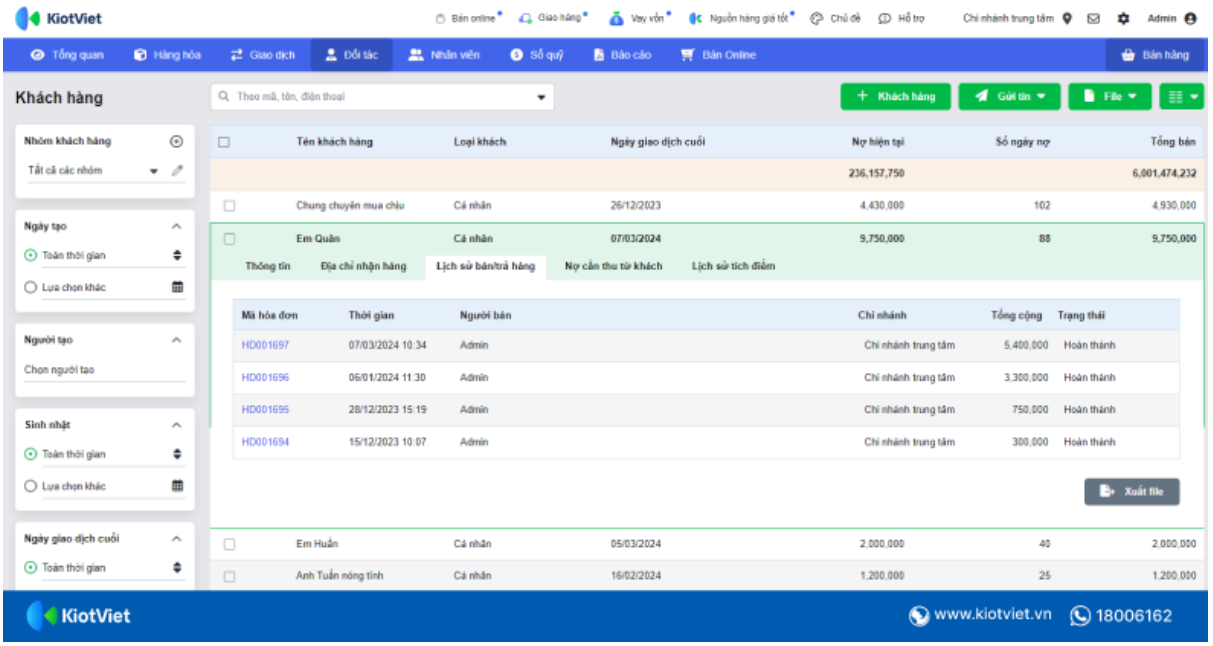
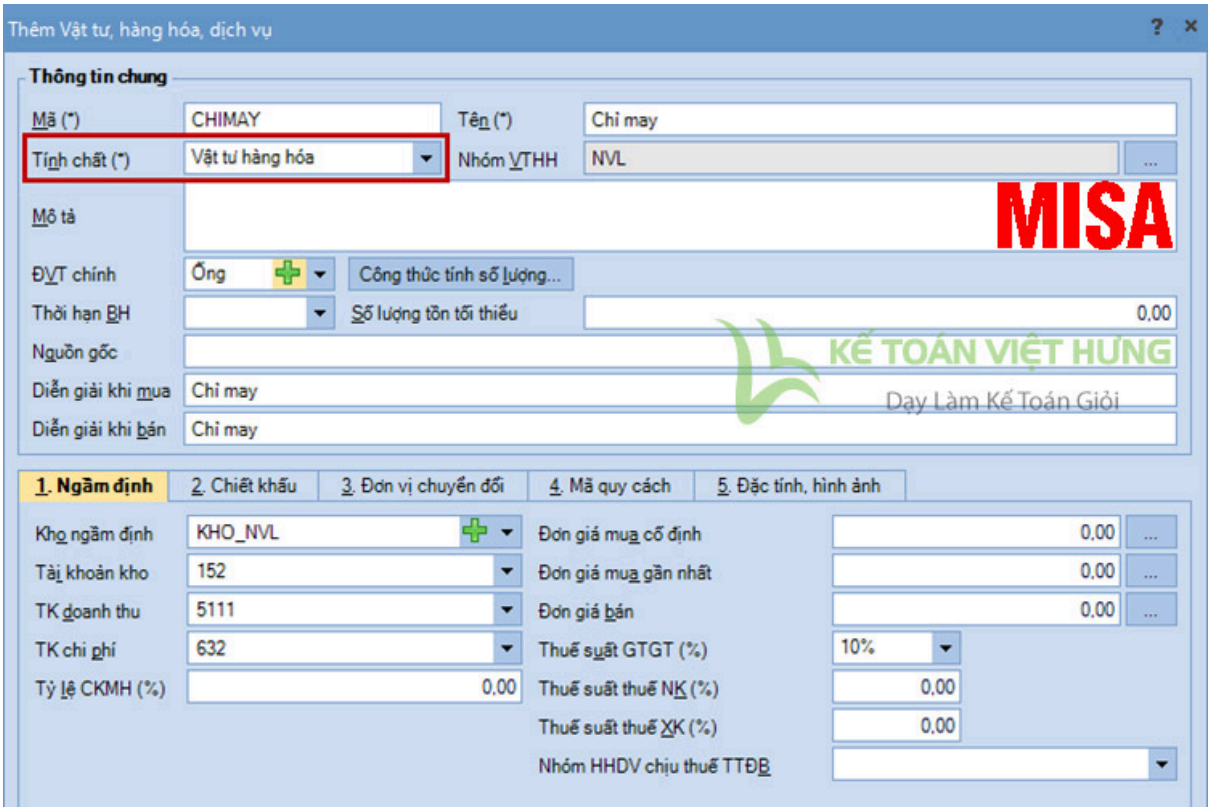
Họ và tên: Trịnh Khánh Huyền
MSSV: 31221026063
Lớp: Phát triển ứng dụng Desktop sáng thứ 4 - B2.511

1. Khi mở phần mềm MISA, KiotViet, hoặc phần mềm quản lý tiệm tóc, thấy những control nào xuất hiện nhiều nhất? Đâu là điểm mạnh của WinForms so với các framework hiện đại (ví dụ tốc độ, triển khai nội bộ, chi phí)?



Phần mềm KiotViet



Phần mềm MISA

- Khi mở các phần mềm này, em thấy các control xuất hiện nhiều nhất là:

Label: tên/mô tả cho các control khác, để người dùng nhận dạng được cần phải làm gì tại ô này
Textbox: ô dùng để nhập thông tin (ví dụ như mã, tên sản phẩm…)
Button: nút dùng để thao tác xuất file, lưu, sửa, xóa,...
ComboBox: chọn một giá trị từ danh sách có sẵn (ví dụ như tính chất, kho ngầm định…)
DataGridView: hiển thị danh sách thông tin (ví dụ như thông tin các giao dịch bán hàng…)
TabControl: chia các nội dung thông tin thành các tab riêng biệt

- Điểm mạnh của WinForms so với các framework hiện đại (ví dụ tốc độ, triển khai nội bộ, chi phí)

Tốc độ: WinForms có tốc độ xử lý dữ liệu nhanh chóng, tải nhanh và có hiệu suất xử lý dữ liệu tốt, đặc biệt với các ứng dụng quản lý, nhập liệu, tính toán, thống kê đơn giản hoặc có cấu trúc không quá phức tạp. Là một công nghệ đã được phát triển, sử dụng và có nhiều cải tiến trong thời gian dài, WinForms mang lại hiệu suất ổn định cho các ứng dụng.
Triển khai nội bộ: Ứng dụng WinForms thường được triển khai dưới dạng ứng dụng máy tính để bàn, giúp việc cài đặt, quản lý dữ liệu nội bộ, kết nối với các thiết bị ngoại

vi trong phạm vi nội bộ trở nên đơn giản và trực tiếp hơn. WinForms nhẹ hơn các framework sử dụng đồ họa phức tạp nên có thể chạy tốt trên các máy tính cũ hoặc có cấu hình yếu, phù hợp với các doanh nghiệp vừa và nhỏ.
Chi phí: WinForms sử dụng mô hình lập trình đơn giản và công cụ thiết kế kéo thả rất trực quan, đồng thời do đã phát triển lâu đời với một cộng đồng lớn và kho thư viện control của bên thứ ba rất đa dạng để xây dựng giao diện quản lý nhanh chóng và chuyên nghiệp, từ đó giúp giảm thời gian và chi phí phát triển cho các ứng dụng có giao diện chuẩn mực, không yêu cầu đồ họa cao cấp.

2. Khi thiết kế form cho màn hình nhỏ (máy tính cũ, độ phân giải thấp), bạn sẽ ưu tiên layout nào? Nếu bạn được giao thiết kế phần mềm cho trường đại học hoặc cửa hàng của gia đình — bạn sẽ bắt đầu từ đâu?

Đối với form cho màn hình nhỏ (máy tính cũ, độ phân giải thấp) em sẽ ưu tiên FlowLayoutPanel vì panel này tự động sắp xếp control theo hàng ngang hoặc dọc, khi form hoặc cửa sổ co giãn, control tự di chuyển xuống dòng hoặc sang phải, không bao giờ yêu cầu cuộn ngang, đảm bảo trải nghiệm người dùng tốt nhất trên không gian giới hạn. Nếu em được giao thiết kế phần mềm cho trường đại học hoặc cửa hàng của gia đình, em sẽ bắt đầu từ

- Phân tích yêu cầu và quy trình nghiệp vụ: cần xác định mục tiêu của phần mềm là gì, thu hẹp phạm vi và chọn một module nhỏ cụ thể trước để dễ quản lý hơn. Phân tích người dùng và các tác vụ cần thiết phục vụ cho người dùng đó.
- Thiết kế cơ sở dữ liệu cho phần mềm để chứa dữ liệu tương tác
- Thiết kế giao diện người dùng: vẽ phác thảo các màn hình chính để xác định với những tác vụ đó thì nên sử dụng control gì, có các form gì, sau đó code để tạo form và các chức năng cơ bản cần có cho phần mềm (vd như thêm, sửa, xóa…), cuối cùng là kết nối cơ sở dữ liệu và kiểm thử phần mềm trước khi sử dụng chính thức.

3. Tạo form đăng ký sản phẩm

Dùng DataTable để khởi tạo CSDL giả lập chứa dữ liệu người dùng nhập vào, CSDL này sẽ làm nguồn dữ liệu cho DataGridView

```
namespace BT3
{
    4 references
    public partial class BT3 : Form
    {
        //Khai báo DataTable để lưu trữ dữ liệu sản phẩm
        private DataTable dtSanPham = new DataTable();
        1 reference
        public BT3()
        {
            InitializeComponent();
            // Gọi hàm khởi tạo CSDL giả lập
            SetupDataTable();
        }
        1 reference
        private void SetupDataTable()
        {
            dtSanPham.Columns.Add("TenSP", typeof(string));
            dtSanPham.Columns.Add("LoaiSP", typeof(string));
            dtSanPham.Columns.Add("SoLuong", typeof(int));

            // Gán DataTable làm nguồn dữ liệu cho DataGridView
            dgvSanPham.DataSource = dtSanPham;
        }
```

Xử lý các yêu cầu của bài tập cho nút Lưu, gán các giá trị từ controls vào các cột của DataRow

```
1 reference
private void btnLuu_Click(object sender, EventArgs e)
{
    if (string.IsNullOrEmpty(txtTenSP.Text) || cboLoaiSP.SelectedItem == null)
    {
        MessageBox.Show("Vui lòng nhập Tên sản phẩm và chọn Loại sản phẩm.", "Lỗi nhập liệu", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }

    DataRow newRow = dtSanPham.NewRow();

    //Gán giá trị từ controls vào các cột của DataRow
    newRow["TenSP"] = txtTenSP.Text;
    newRow["LoaiSP"] = cboLoaiSP.SelectedItem.ToString();
    newRow["SoLuong"] = (int)nudSoLuong.Value;

    dtSanPham.Rows.Add(newRow);

    MessageBox.Show("Đã thêm sản phẩm thành công!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);
    btnLamMoi_Click(sender, e); // Tái sử dụng code Làm mới
}
```

Xử lý các yêu cầu cho nút Làm mới

```
2 references
private void btnLamMoi_Click(object sender, EventArgs e)
{
    txtTenSP.Clear();
    cboLoaiSP.SelectedIndex = -1; // Bỏ chọn ComboBox
    nudSoLuong.Value = nudSoLuong.Minimum; // Đặt lại số lượng về giá trị nhỏ nhất
    txtTenSP.Focus();
}
```

Xử lý các yêu cầu cho nút Xóa

```
1 reference
private void btnXoa_Click(object sender, EventArgs e)
{
    //Kiểm tra xem có dòng nào được chọn không
    if (dgvSanPham.SelectedRows.Count > 0)
    {
        //Lấy chỉ số của dòng đầu tiên được chọn
        int rowIndex = dgvSanPham.SelectedRows[0].Index;

        DialogResult confirm = MessageBox.Show("Bạn có chắc chắn muốn xóa sản phẩm này?", "Xác nhận Xóa", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

        if (confirm == DialogResult.Yes)
        {
            //Dùng Rows.RemoveAt vì dgvSanPham.Rows có thể chứa các dòng không phải DataRow (ví dụ: dòng thêm mới)
            dtSanPham.Rows.RemoveAt(rowIndex);
            MessageBox.Show("Đã xóa sản phẩm thành công.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
    }
    else
    {
        MessageBox.Show("Vui lòng chọn một dòng sản phẩm để xóa.", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Warning);
    }
}
```

Người dùng khi sử dụng sẽ nhập tên sản phẩm, chọn loại sản phẩm, số lượng có thể nhập tay hoặc dùng nút tăng giảm, nhấn các nút để thao tác. Bấm nút lưu dữ liệu để dữ liệu hiển thị trên DataGridView, khi cần xóa một dòng dữ liệu, người dùng bấm chuột vào dòng dữ liệu và bấm xóa, cuối cùng là nút làm mới lại bảng dữ liệu sau khi thêm và xóa.

4. Tạo form đăng nhập cho bài tập 3

Kiểm tra lỗi đăng nhập của người dùng, hiển thị lỗi thông qua ErrorProvider

```
1 reference
public LoginForm()
{
    InitializeComponent();
}
1 reference
private bool KiemTraLoi()
{
    bool isError = false;

    //Kiểm tra Username
    if (string.IsNullOrEmpty(txtUsername.Text))
    {
        errorProvider1.SetError(txtUsername, "Tên đăng nhập không được để trống!");
        isError = true;
    }
    else
    {
        errorProvider1.SetError(txtUsername, ""); // Xóa lỗi nếu đã nhập
    }

    //Kiểm tra Password
    if (string.IsNullOrEmpty(txtPassword.Text))
    {
        errorProvider1.SetError(txtPassword, "Mật khẩu không được để trống!");
        isError = true;
    }
    else
    {
        errorProvider1.SetError(txtPassword, "");
    }

    return isError;
}
```

Thiết lập cho nút Đăng nhập, hiển thị thông báo nếu đăng nhập thành công và thất bại. Khi đăng nhập thành công sẽ ẩn form đăng nhập để chuyển qua form "Đăng ký sản phẩm" ở bài 3.

```
1 reference
private void btnLogin_Click(object sender, EventArgs e)
{
    //Kiểm tra và thoát nếu có lỗi để trống
    if (KiemTraLoi())
    {
        return;
    }

    string username = txtUsername.Text;
    string password = txtPassword.Text;

    //Đăng nhập thành công
    if (username == "admin" && password == "123")
    {
        MessageBox.Show("Đăng nhập thành công!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);

        BT3 mainForm = new BT3();
        mainForm.Show();

        // Ẩn Form đăng nhập hiện tại
        this.Hide();
    }
    //Đăng nhập thất bại
    else
    {
        MessageBox.Show("Sai tài khoản hoặc mật khẩu!", "Lỗi đăng nhập", MessageBoxButtons.OK, MessageBoxIcon.Error);
        txtPassword.Clear(); // Xóa mật khẩu cũ
        txtPassword.Focus();
    }
}
```

Vì yêu cầu form đăng nhập phải hiển thị lên trước form “Đăng ký sản phẩm” nên cần thiết lập ở file Program.cs

```
namespace BT3
{
    0 references
    internal static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        0 references
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new LoginForm());
        }
    }
}
```

Người dùng thực hiện nhập tài khoản và mật khẩu, nếu đúng sẽ được chuyển qua form “Đăng ký sản phẩm” để tiếp tục thao tác. Trong trường hợp không biết cần nhập gì, rê chuột đến ô tài khoản và mật khẩu để nhận được gợi ý, nếu sai sẽ có thông báo lỗi hiện lên.

5. Tạo Form “Quản lý sản phẩm cửa hàng Việt Nam”.

Sử dụng BindingList vì nó tự động cập nhật DataGridView và khai báo thêm một biến
Xóa nội dung các ô nhập liệu sau khi đã thêm dữ liệu vào

```
5 references
public class Product
{
    2 references
    public string Ten { get; set; }
    2 references
    public string Loai { get; set; }
    2 references
    public int SoLuong { get; set; }
    2 references
    public string TinhTrang { get; set; } // "Còn hàng" hoặc "Hết hàng"
}
private BindingList<Product> productList = new BindingList<Product>();
1 reference
public QuanLySanPhamForm()
{
    InitializeComponent();
    dgvSanPham.DataSource = productList; // Gán nguồn dữ liệu
    rdoConHang.Checked = true; // Mặc định chọn Còn hàng
}
2 references
private void ClearControls()
{
    // Xóa nội dung của các ô nhập liệu
    txtTenSP.Clear();
    cboLoaiSP.SelectedIndex = -1; // Bỏ chọn ComboBox
    nudSoLuong.Value = nudSoLuong.Minimum; // Đặt về giá trị mặc định
    rdoConHang.Checked = true; // Mặc định chọn lại "Còn hàng"

    // Xóa tất cả các thông báo lỗi
    errorProvider1.Clear();

    txtTenSP.Focus();
}
```

Kiểm tra các lỗi khi nhập thông tin và xử lý các sự kiện được yêu cầu của nút Thêm

```
1 reference
private void btnThem_Click(object sender, EventArgs e)
{
    // Kiểm tra lỗi (Yêu cầu d)
    if (string.IsNullOrEmpty(txtTenSP.Text))
    {
        errorProvider1.SetError(txtTenSP, "Tên sản phẩm không được trống!");
        return;
    }
    if (nudSoLuong.Value <= 0)
    {
        errorProvider1.SetError(nudSoLuong, "Số lượng phải lớn hơn 0!");
        return;
    }

    // Xóa lỗi nếu nhập đúng
    errorProvider1.Clear();

    Product newProduct = new Product
    {
        Ten = txtTenSP.Text,
        Loai = cboLoaiSP.SelectedItem?.ToString() ?? "Chưa phân loại",
        SoLuong = (int)nudSoLuong.Value,
        // Xác định tình trạng sản phẩm
        TinhTrang = rdoConHang.Checked ? "Còn hàng" : "Hết hàng"
    };

    productList.Add(newProduct);

    // Cập nhật StatusStrip (Yêu cầu e.ii)
    lblTongSanPham.Text = $"Tổng số sản phẩm: {productList.Count}";

    // Xóa dữ liệu nhập để tiếp tục
    txtTenSP.Clear();
    nudSoLuong.Value = nudSoLuong.Minimum;
    txtTenSP.Focus();
    ClearControls();
```

Xử lý nút Xóa

```
1 reference
private void btnXoa_Click(object sender, EventArgs e)
{
    if (dgvSanPham.SelectedRows.Count > 0)
    {
        // Lấy chỉ số của đối tượng trong List
        int index = dgvSanPham.SelectedRows[0].Index;

        // Xác nhận xóa
        DialogResult confirm = MessageBox.Show("Xác nhận xóa sản phẩm này?", "Xóa", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

        if (confirm == DialogResult.Yes)
        {
            productList.RemoveAt(index);
            lblTongSanPham.Text = $"Tổng số sản phẩm: {productList.Count}"; // Cập nhật lại
        }
    }
    else
    {
        MessageBox.Show("Vui lòng chọn dòng cần xóa.", "Thông báo");
    }
```

Để thực hiện chức năng Sửa, cần tải dữ liệu của dòng đang chọn trong DataGridView lên các ô nhập liệu, vì vậy cần sử dụng Cell Click để tạo hàm xử lý

```
private int selectedProductIndex = -1;
1 reference
private void dgvSanPham_CellClick(object sender, DataGridViewCellEventArgs e)
{
    // Đảm bảo người dùng click vào một hàng hợp lệ (không phải hàng header)
    if (e.RowIndex >= 0)
    {
        DataGridViewRow row = dgvSanPham.Rows[e.RowIndex];

        //Tải dữ liệu lên các controls
        txtTenSP.Text = row.Cells["ColTenSP"].Value.ToString();
        cboLoaiSP.SelectedItem = row.Cells["ColLoaiSP"].Value.ToString();
        nudSoLuong.Value = Convert.ToInt32(row.Cells["ColSoLuong"].Value);

        string tinhTrang = row.Cells["ColTinhTrang"].Value.ToString();
        if (tinhTrang == "Còn hàng")
        {
            rdoConHang.Checked = true;
        }
        else
        {
            rdoHetHang.Checked = true;
        }
    }
    else
    {
        // Nếu click vào Header (RowIndex < 0)
        selectedProductIndex = -1;
    }
}
```

Thực hiện xử lý các thao tác với nút Sửa, cần kiểm tra lỗi nhập liệu và xem thử người dùng đã chọn dòng nào để sửa chưa

```
    private void btnSua_Click(object sender, EventArgs e)
    {
        //Kiểm tra xem đã chọn dòng nào để sửa chưa
if (selectedProductIndex < 0 || selectedProductIndex >= productList.Count)
        {
            MessageBox.Show("Vui lòng chọn một sản phẩm từ danh sách để sửa.", "Lỗi Sửa", MessageBoxButtons.OK, MessageBoxIcon.Warning);
            return;
        }

        //Kiểm tra lỗi nhập liệu trước khi sửa (Tái sử dụng code kiểm tra lỗi nếu cần)
        if (string.IsNullOrEmpty(txtTenSP.Text))
        {
            errorProvider1.SetError(txtTenSP, "Tên sản phẩm không được trống!");
            return;
        }
        if (nudSoLuong.Value <= 0)
        {
            errorProvider1.SetError(nudSoLuong, "Số lượng phải lớn hơn 0!");
            return;
        }
        errorProvider1.Clear(); // Xóa lỗi nếu dữ liệu hợp lệ

        //Lấy đối tượng Product cần sửa từ List
        Product productToEdit = productList[selectedProductIndex];

        //Cập nhật các thuộc tính của đối tượng
        productToEdit.Ten = txtTenSP.Text;
        productToEdit.Loai = cboLoaiSP.SelectedItem?.ToString() ?? "Chưa phân loại";
        productToEdit.SoLuong = (int)nudSoLuong.Value;
        productToEdit.TinhTrang = rdoConHang.Checked ? "Còn hàng" : "Hết hàng";

        //Thông báo và làm mới giao diện
        MessageBox.Show("Sản phẩm đã được cập nhật thành công!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);

        // Đặt lại index về -1 để tránh sửa nhầm lần sau
        selectedProductIndex = -1;

        // Gọi hàm làm sạch controls
        ClearControls();
```

Khi dùng Form, người dùng nhập tên sản phẩm, chọn loại sản phẩm, điền số lượng hoặc nhấn nút tăng giảm, chọn tình trạng sản phẩm, sau đó bấm lưu, sửa hoặc xóa. Với nút sửa và xóa, cần chọn dòng rồi mới thực hiện thao tác.