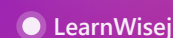


Navigation réutilisable et permissions

Les quatre vues que vous allez construire, un shell unique qui possède la navigation et un premier aperçu des permissions.



Objectif de la leçon

Découvrir la cible que vous construirez dans le lab : un shell réutilisable dont la zone de contenu alterne entre quatre vues — Dashboard, Tickets, Customers et Settings — tandis que l'en-tête, la navigation et la barre d'état restent identiques.

1. Les quatre vues

Le shell extérieur ne change jamais. Chaque bouton de navigation charge un **UserControl** différent dans le panneau de contenu.

Vue	Ce qu'elle affiche
DashboardView	Des cartes de synthèse (Open Tickets, Customers, System Status) et une action rapide comme <i>Run Health Check</i> .
TicketsView	Une liste de tickets de support avec les actions <i>New Ticket</i> et <i>Close Selected</i> .
CustomersView	Une liste de clients avec les actions <i>Add Customer</i> et <i>View Selected</i> .
SettingsView	Le rôle courant et un bouton <i>Save Settings</i> qui introduit la gestion des permissions.



Pourquoi des UserControls

Placer chaque page dans son propre UserControl garde l'interface répétée organisée et réutilisable, et permet au shell de rester léger — il ne possède que la navigation et le panneau de contenu.

2. Un seul shell possède la navigation

Les noms exacts peuvent varier, mais l'idée reste la même : le shell crée la bonne vue, la docke pour remplir le panneau de contenu, puis met à jour le fil d'Ariane et l'état.

```
// Main shell owns the navigation buttons and the content panel.
public partial class MainPage : Page
{
    private string currentRole = "Support Agent";

    public MainPage()
    {
        InitializeComponent();
        NavigateTo("Dashboard");
    }
}
```

```

        ApplyPermissions();
    }

    private void NavigateTo(string pageName)
    {
        pnlContent.Controls.Clear();

        UserControl view;
        switch (pageName)
        {
            case "Tickets":    view = new TicketsView();    break;
            case "Customers":  view = new CustomersView(); break;
            case "Settings":   view = new SettingsView(currentRole); break;
            default:           view = new DashboardView(); break;
        }

        view.Dock = DockStyle.Fill;
        pnlContent.Controls.Add(view);
        lblBreadcrumb.Text = "Home / " + pageName;
        lblStatus.Text = DateTime.Now.ToString("hh:mm:ss tt") + " - Opened " + pageName + "
page.";
    }

    private void ApplyPermissions()
    {
        // Beginner version: a Support Agent can view Settings but not save.
        // Later this grows into real authentication and authorization.
    }
}

```

3. Des paramètres en lecture seule — la première permission

La vue Settings est l'endroit où les permissions deviennent visibles. Pour un **Support Agent**, affichez les paramètres mais désactivez l'enregistrement — le bouton *Save Settings* est grisé et accompagné d'une note comme « *Settings are view-only for this role.* » Un Manager, lui, aurait le droit d'enregistrer. Cela introduit les permissions **avant** même qu'un système de connexion existe.

4. Liste de contrôle dans le Designer

- Utilisez `Dock` ou `Anchor` sur l'en-tête, la navigation à gauche, le panneau de contenu et la barre d'état.
- Supprimez le bouton de réduction d'une Window/Form1, sauf si vous construisez une zone de barre des tâches/restauration.
- Préférez une **Page** lorsque le shell doit remplir le navigateur comme une application web classique.
- Centralisez la logique de navigation dans le shell au lieu de la dupliquer dans chaque page.
- Utilisez des UserControls pour le contenu répété — `DashboardView`, `TicketsView`, `CustomersView`, `SettingsView`.



Vous êtes prêt pour le lab

Ensuite : regardez la vidéo pas à pas du shell de navigation, réussissez la vérification des connaissances, puis construisez le shell de navigation réutilisable — en-tête, quatre boutons de navigation, une zone de contenu interchangeable, un journal d'état et des paramètres en lecture seule pour un Support Agent.