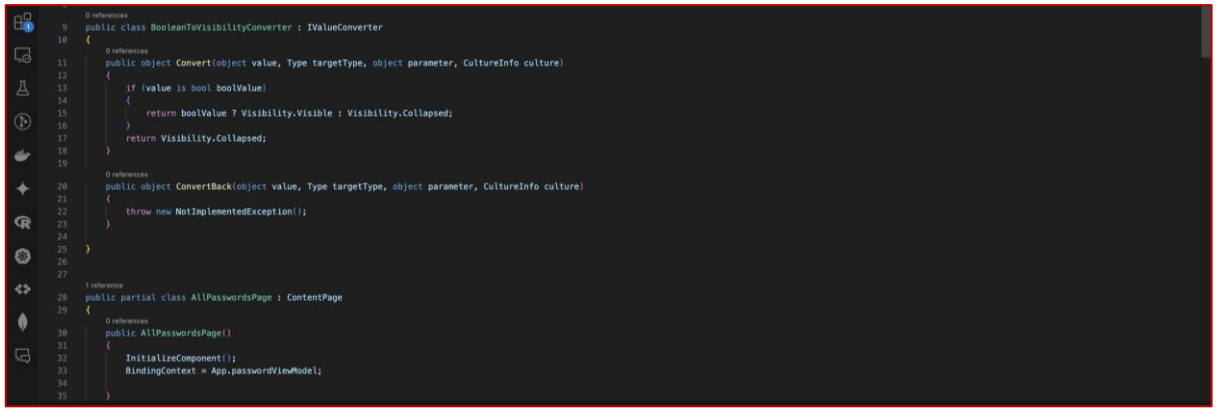
GVCIS641.01.202510

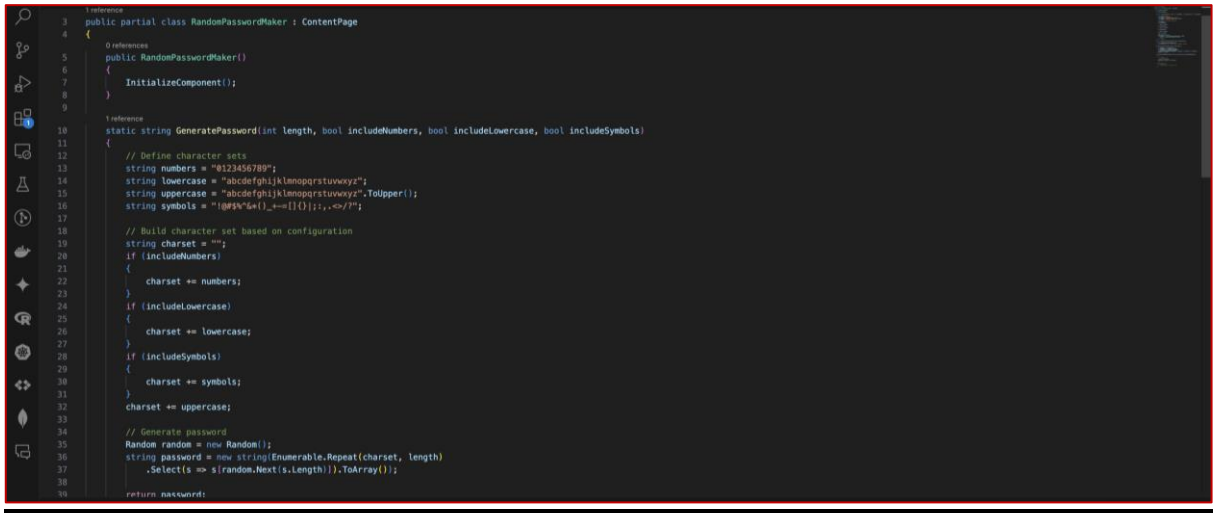
CIS 641 01 - IP - Systems Analysis and Design (F24)

1.All Password Store Page:



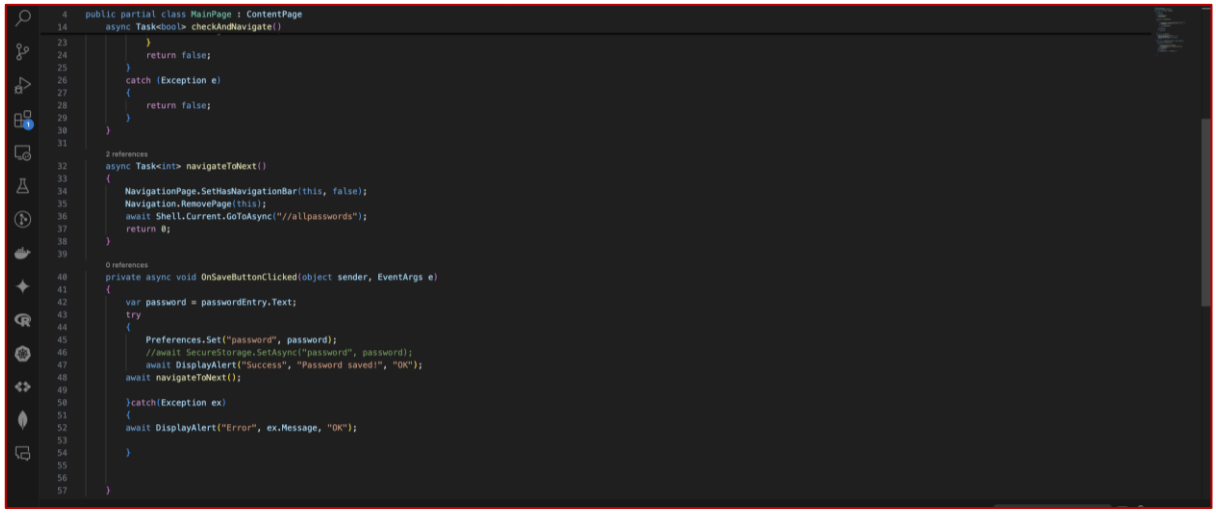
```
0 references
public class BooleanToVisibilityConverter : IValueConverter
{
    0 references
    public object Convert(object value, Type targetType, object parameter, CultureInfo culture)
    {
        if (value is bool boolValue)
        {
            return boolValue ? Visibility.Visible : Visibility.Collapsed;
        }
        return Visibility.Collapsed;
    }

    0 references
    public object ConvertBack(object value, Type targetType, object parameter, CultureInfo culture)
    {
        throw new NotImplementedException();
    }

}


1 reference
public partial class AllPasswordsPage : ContentPage
{
    0 references
    public AllPasswordsPage()
    {
        InitializeComponent();
        BindingContext = App.passwordViewModel;

    }
```

2.Random Password Genertor Page:



```
1 reference
public partial class RandomPasswordMaker : ContentPage
{
    0 references
    public RandomPasswordMaker()
    {
        InitializeComponent();
    }

    1 reference
    static string GeneratePassword(int length, bool includeNumbers, bool includeLowercase, bool includeSymbols)
    {
        // Define character sets
        string numbers = "0123456789";
        string lowercase = "abcdefghijklmnopqrstuvwxyz";
        string uppercase = "abcdefghijklmnopqrstuvwxyz".ToUpper();
        string symbols = "!@#$%^&*()_+-=[]{}|;:,.<>/?";

        // Build character set based on configuration
        string charset = "";
        if (includeNumbers)
        {
            charset += numbers;
        }
        if (includeLowercase)
        {
            charset += lowercase;
        }
        if (includeSymbols)
        {
            charset += symbols;
        }
        charset += uppercase;

        // Generate password
        Random random = new Random();
        string password = new string(Enumerable.Repeat(charset, length)
            .Select(s => s[random.Next(s.Length)]).ToArray());

        return password;
```

3.Password Store Alert Page :



```
public partial class MainPage : ContentPage
    async Task<bool> checkAndNavigate()
            }
            return false;
        }
        catch (Exception e)
        {
            return false;
        }
    }

    2 references
    async Task<int> navigateToNext()
    {
        NavigationPage.SetHasNavigationBar(this, false);
        Navigation.RemovePage(this);
        await Shell.Current.GoToAsync("//allpasswords");
        return 0;
    }

    0 references
    private async void OnSaveButtonClicked(object sender, EventArgs e)
    {
        var password = passwordEntry.Text;
        try
        {
            Preferences.Set("password", password);
            //await SecureStorage.SetAsync("password", password);
            await DisplayAlert("Success", "Password saved!", "OK");
        await navigateToNext();

        }catch(Exception ex)
        {
        await DisplayAlert("Error", ex.Message, "OK");

        }


    }
```

4.Adding New Application Page:

```
1 reference
public partial class NewApplication : ContentPage
{
    0 references
    public NewApplication()
    {
        InitializeComponent();
    }

    0 references
    async void SaveButton_Clicked(object sender, EventArgs e)
    {
        try
        {
            var passwordviewModel = App.passwordViewModel;
            string appName = appNameEntry.Text;
            string appPassword = appPasswordEntry.Text;

            passwordviewModel.saveToHistory(appName, appPassword);
            await DisplayAlert("Alert", "Entry added to password manager", "OK");
        }
        catch(Exception ex)
        {
            await DisplayAlert("Alert", ex.Message, "OK");

        }

    }
}
```