**1914626\_ Nguyễn Trung Nguyên**

**Bài tập tuần 3**

**Program.CS**

static class Program

{

/// <summary>

/// The main entry point for the application.

/// </summary>

[STAThread]

static void Main()

{

INewRepository repository = new NewsRepository();

NewParser parser = new NewParser();

RssReader reader = new RssReader(parser);

var manager = new NewsFeedManager(repository, reader);

Application.SetHighDpiMode(HighDpiMode.SystemAware);

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new MainForm(manager));

}

}

**NewsFeedManager.cs**

public class NewsFeedManager

{

private readonly INewRepository \_newRepository;

private List<Publisher> \_publishers;

private readonly RssReader \_rssReader;

public NewsFeedManager(INewRepository newRepository, RssReader rssReader)

{

\_newRepository = newRepository;

\_rssReader = rssReader;

}

public List<Publisher> GetNewsFeed()

{

if(\_publishers == null)

{

\_publishers = \_newRepository.GetNews();

}

return \_publishers;

}

public void SaveChanges()

{

\_newRepository.Save(\_publishers);

}

public void RemovePublisher(string publisherName)

{

\_publishers.RemoveAll(x => x.Name == publisherName);

SaveChanges();

}

public void RemoveCategory(string \_publisherName, string categoryName)

{

var publisher = \_publishers.Find(x => x.Name == \_publisherName);

if (publisher == null)

return;

publisher.RemoveCategory(categoryName);

SaveChanges();

}

public bool AddCategory(string publisherName, string categoryName, string rssLink, bool updateIfExisted)

{

var publisher = \_publishers.Find(x => x.Name == publisherName);

if (publisher == null)

{

publisher = new Publisher()

{

Name = publisherName

};

\_publishers.Add(publisher);

}

return publisher.AddCategory(categoryName, rssLink, updateIfExisted);

}

public List<Article> GetNews(string publisherName, string categoryName)

{

var publisher = \_publishers.Find(x => x.Name == publisherName);

if (publisher == null) return new List<Article>();

var category = publisher.Categories.Find(x => x.Name == categoryName);

if (category == null) return new List<Article>();

if(category.Articles.Count == 0)

{

category.Articles = \_rssReader.GetNews(category.RssLink);

}

return category.Articles;

}

}

**Form1.CS**

public partial class MainForm : Form

{

private readonly NewsFeedManager \_newsManager;

public MainForm(NewsFeedManager newsManager)

{

\_newsManager = newsManager;

InitializeComponent();

}

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

{

var dialog = new AddFeedForm(\_newsManager);

dialog.ShowDialog(this);

if (dialog.HasChanges)

{

\_newsManager.SaveChanges();

ShowFeedOnTreeView(\_newsManager.GetNewsFeed());

}

}

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

{

if (tvwPublisher.SelectedNode == null) return;

if(tvwPublisher.SelectedNode.Level == 0)

{

\_newsManager.RemovePublisher(tvwPublisher.SelectedNode.Text);

}

else

{

var publisherNode = tvwPublisher.SelectedNode.Parent;

\_newsManager.RemoveCategory(publisherNode.Text, tvwPublisher.SelectedNode.Text);

}

tvwPublisher.SelectedNode.Remove();

}

private void tvwPublisher\_AfterSelect(object sender, TreeViewEventArgs e)

{

pnlNews.Controls.Clear();

if(e.Node.Level == 1)

{

var articles = \_newsManager.GetNews(e.Node.Parent.Text, e.Node.Text);

foreach (var article in articles)

{

var item = new NewsControl();

item.Size = new Size(500, 100);

item.Dock = DockStyle.Top;

item.SetArticle(article);

pnlNews.Controls.Add(item);

}

}

}

public void ShowFeedOnTreeView(List<Publisher> publishers)

{

tvwPublisher.Nodes.Clear();

pnlNews.Controls.Clear();

foreach(var publisher in publishers)

{

var publisherNode = tvwPublisher.Nodes.Add(publisher.Name);

foreach(var category in publisher.Categories)

{

publisherNode.Nodes.Add(category.Name);

}

}

tvwPublisher.ExpandAll();

}

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

{

ShowFeedOnTreeView(\_newsManager.GetNewsFeed());

}

}

**AddFeedForm.CS**

public partial class AddFeedForm : Form

{

private readonly NewsFeedManager \_newsManager;

public bool HasChanges { get; set; }

public AddFeedForm(NewsFeedManager newsManager)

{

\_newsManager = newsManager;

InitializeComponent();

}

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

{

var publishers = \_newsManager.GetNewsFeed();

foreach(var publisher in publishers)

{

cbbToaSoan.Items.Add(publisher.Name);

}

}

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

{

var publisherName = cbbToaSoan.Text;

var categoryName = txtChuyenMuc.Text;

var rssLink = txtRssLink.Text;

if(string.IsNullOrWhiteSpace(publisherName)||

string.IsNullOrWhiteSpace(categoryName)||

string.IsNullOrWhiteSpace(rssLink))

{

MessageBox.Show("Bạn phải nhập đầy đủ dữ liệu", "Lỗi");

return;

}

HasChanges = true;

var success = \_newsManager.AddCategory(publisherName, categoryName, rssLink, false);

if (success)

{

ClearForm();

return;

}

if(MessageBox.Show("Chuyên mục này đã tồn tại, bạn có muốn cập nhập RSS link không?","Thông báo", MessageBoxButtons.YesNo) == DialogResult.Yes)

{

\_newsManager.AddCategory(publisherName, categoryName, rssLink, true);

}

ClearForm();

}

private void ClearForm()

{

cbbToaSoan.Text = "";

txtChuyenMuc.Text = "";

txtRssLink.Text = "";

}

**RssReader.CS**

public class RssReader

{

private readonly NewParser \_parser;

public RssReader(NewParser parser)

{

\_parser = parser;

}

public List<Article> GetNews(string rssLink)

{

var feedData = DownloadFeed(rssLink);

return \_parser.ParseXml(feedData);

}

private string DownloadFeed(string rssLink)

{

var client = new WebClient()

{

Encoding = Encoding.UTF8

};

return client.DownloadString(rssLink);

}

}

**NewParser.CS**

public class NewParser

{

public List<Article> ParseXml(string xmlContent)

{

var document = new XmlDocument();

document.LoadXml(xmlContent);

var articles = new List<Article>();

var itemNodes = document.SelectNodes("//item");

foreach (XmlNode node in itemNodes)

{

var news = new Article()

{

Title = node.SelectSingleNode("title").InnerText,

Description = StripHtml(node.SelectSingleNode("description").InnerText),

Link = node.SelectSingleNode("link").InnerText,

PublishedDate = ParseDate(node.SelectSingleNode("pubdate").InnerText)

};

articles.Add(news);

}

return articles;

}

private string StripHtml(string content)

{

return Regex.Replace(content, "<.\*?>", String.Empty).Trim();

}

private DateTime ParseDate(string dateStr)

{

try

{

return DateTime.Parse(dateStr);

}

catch

{

return DateTime.Now;

}

}

}

**Publisher.CS**

public class Publisher

{

public string Name { get; set; }

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

public Publisher()

{

Categories = new List<Category>();

}

public bool AddCategory(string name, string link, bool updateIfExisted)

{

var category = Categories.Find(x => x.Name == name);

if(category == null)

{

category = new Category()

{

Name = name,

RssLink = link

};

Categories.Add(category);

return true;

}

if (updateIfExisted)

{

category.RssLink = link;

return true;

}

return false;

}

public void RemoveCategory(string name)

{

Categories.RemoveAll(x => x.Name == name);

}

}

**NewsRepository.CS**

public class NewsRepository : INewRepository

{

private const string FilePath = "Data\\data.txt";

public List<Publisher> GetNews()

{

var publisher = new List<Publisher>();

Publisher office = null;

string line;

try

{

using (var stream = new FileStream(FilePath,FileMode.Open,FileAccess.Read))

{

using (var reader = new StreamReader(stream))

{

while (!reader.EndOfStream)

{

line = reader.ReadLine();

if(line == null)

{

break;

}

if (line.StartsWith("@"))

{

office = ParsePublisher(line);

publisher.Add(office);

}

else if(line.StartsWith("#") && office != null)

{

var category = ParseCategory(line);

office.Categories.Add(category);

}

}

}

}

}

catch (Exception)

{

throw;

}

return publisher;

}

public void Save(List<Publisher> publishers)

{

using (var stream = new FileStream(FilePath, FileMode.Create, FileAccess.Write))

{

using (var writer = new StreamWriter(stream))

{

foreach (var publisher in publishers)

{

writer.WriteLine("@{0}", publisher.Name);

foreach (var category in publisher.Categories)

{

writer.WriteLine("#{0}^{1}", category.Name, category.RssLink);

}

}

}

}

}

private Publisher ParsePublisher(string info)

{

return new Publisher()

{

Name = info.Substring(1).Trim()

};

}

private Category ParseCategory(string info)

{

var parts = info.Substring(1).Split('^');

return new Category()

{

Name = parts[0].Trim(),

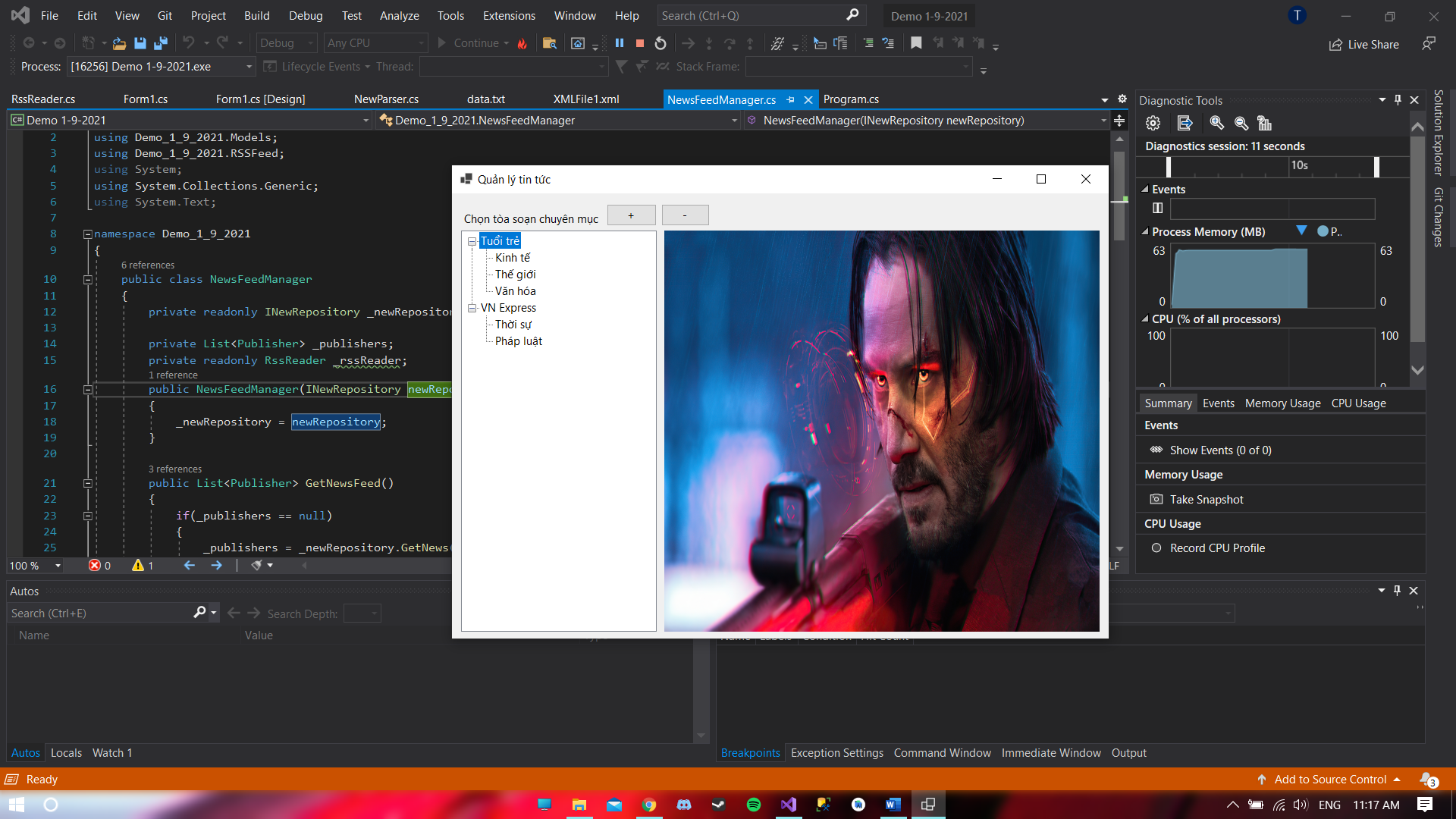
RssLink = parts[1].Trim()

};

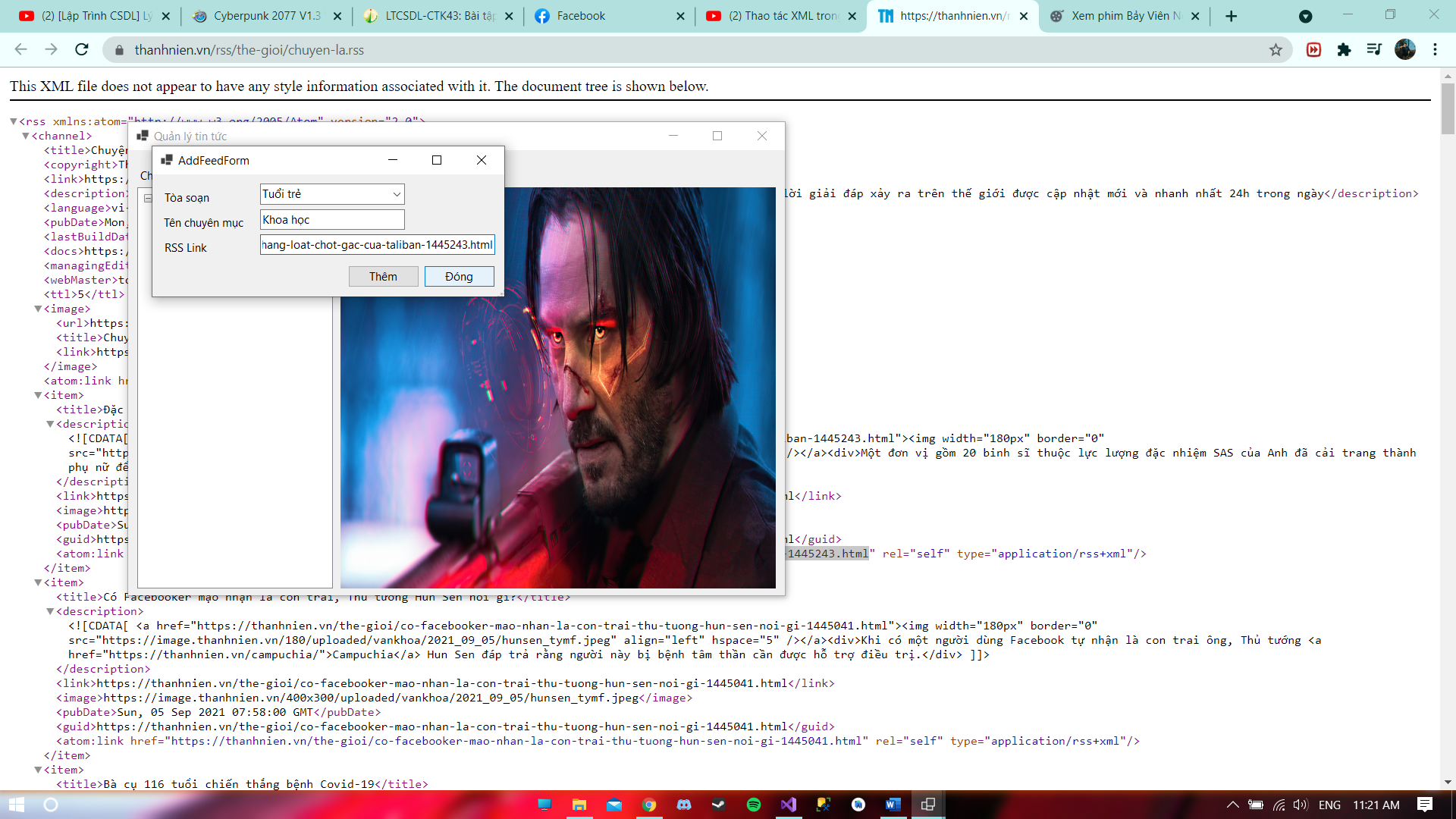
}

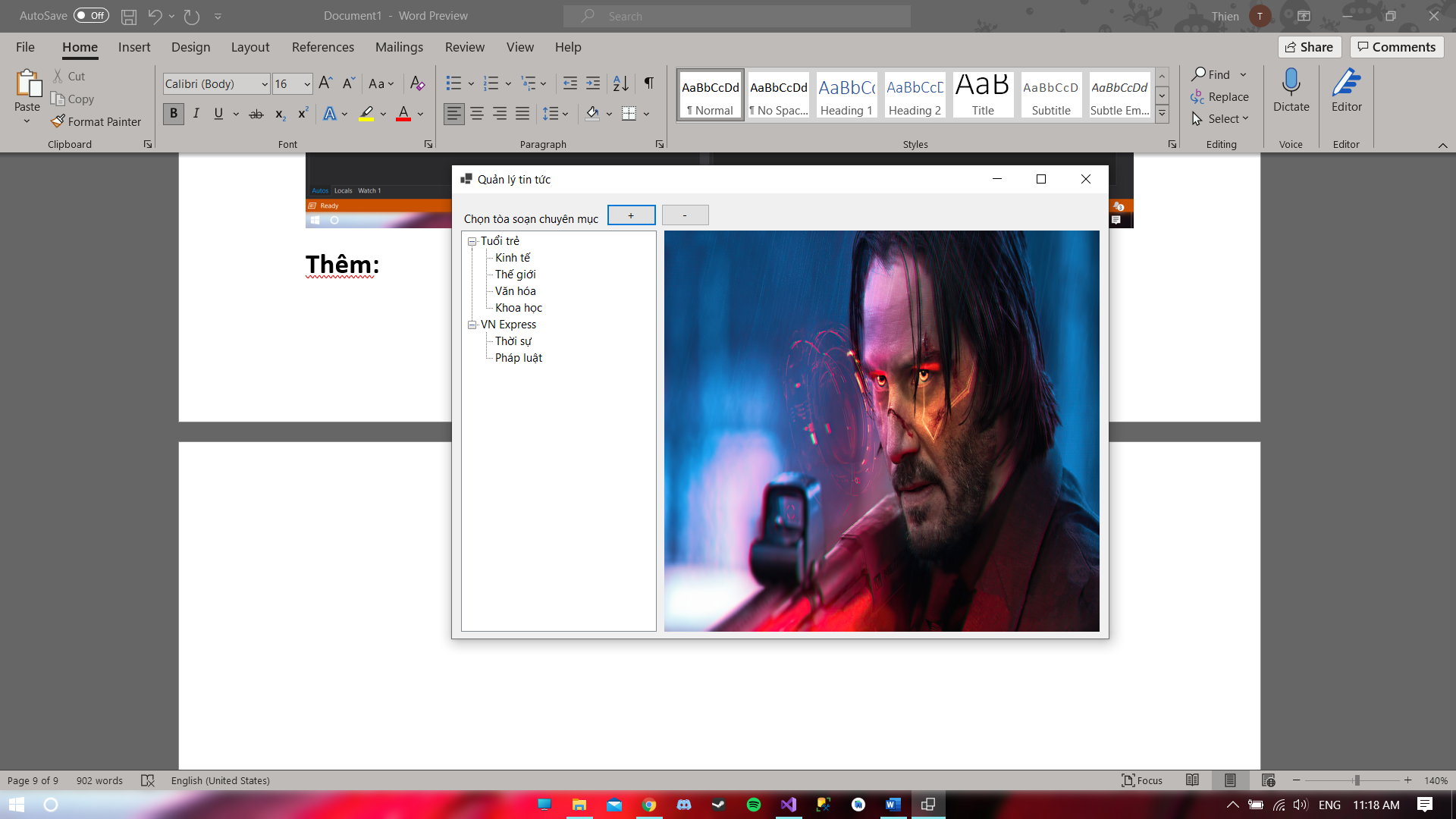
}

**Hình ảnh kết quả:**

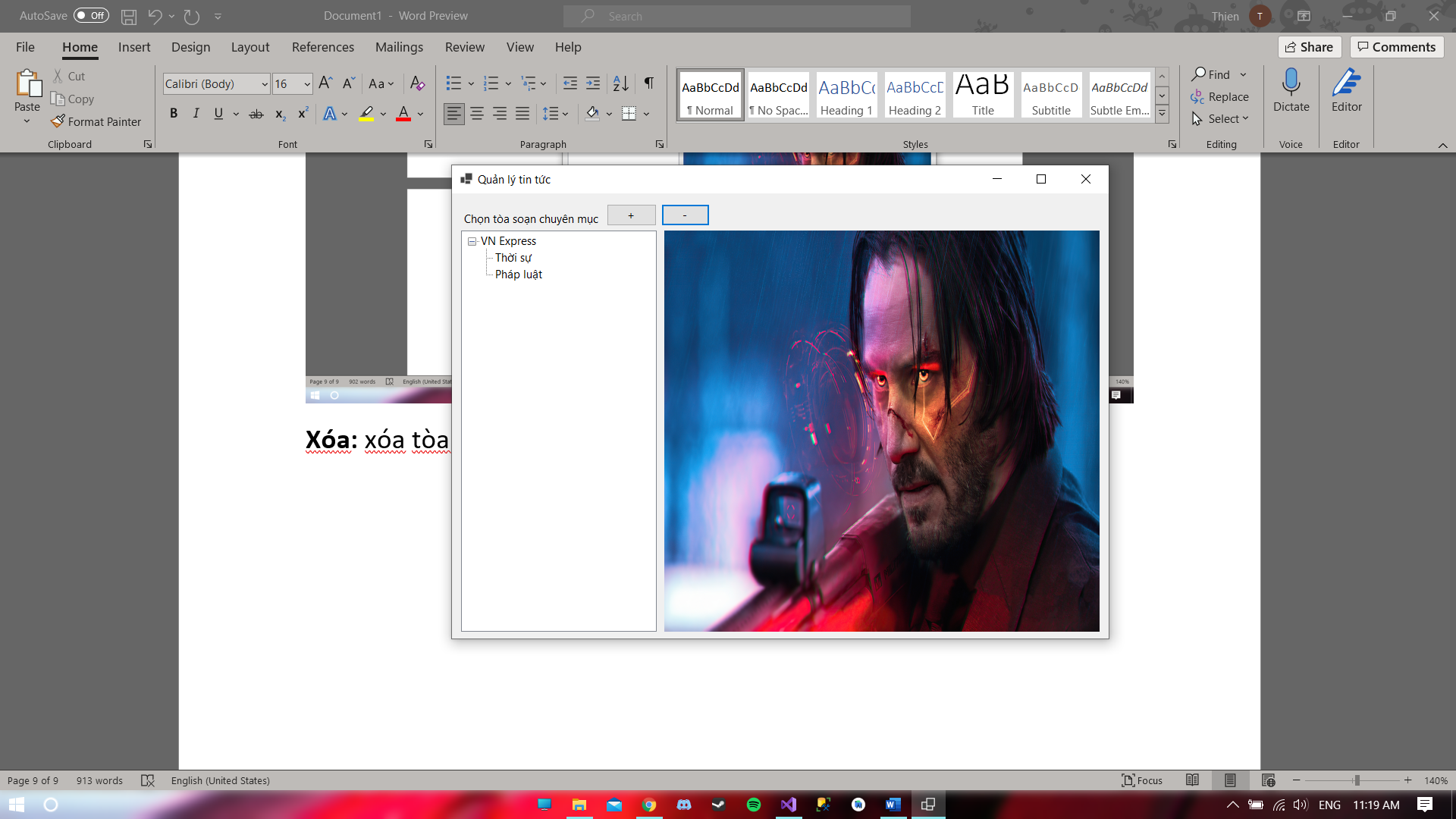


**Thêm:** Thêm chuyên mục Khoa học





**Xóa:** xóa tòa soạn tuổi trẻ



**Hiển thị thông báo bạn chọn**

