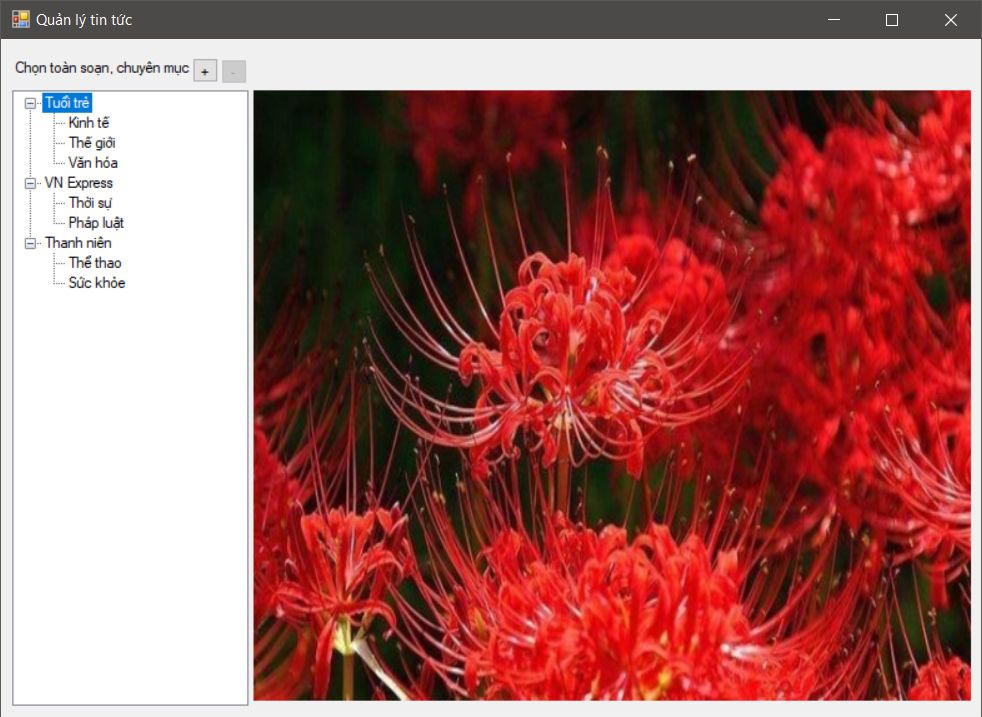
<https://github.com/phhduc/lap_trinh_co_so_du_lieu/tree/main/baitaptuan3/bttuan3>



Graphical user interface, text, application, email

Description automatically generatedGraphical user interface, application

Description automatically generated

Graphical user interface, text, application

Description automatically generatedGraphical user interface, application

Description automatically generated

Graphical user interface, text, application

Description automatically generatedGraphical user interface, application

Description automatically generatedGraphical user interface, text, application

Description automatically generatedA picture containing graphical user interface

Description automatically generated

using bttuan3.RSSFeed;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace bttuan3

{

public class NewsFeedManager

{

private readonly INewsRepository \_newsRepository;

private List<Publisher> \_publishers;

private readonly RssReader \_rssReader;

public NewsFeedManager(INewsRepository newsRepository, RssReader rssReader)

{

\_newsRepository = newsRepository;

\_rssReader = rssReader;

}

public List<Publisher> GetNewsFeed()

{

if (\_publishers == null)

{

\_publishers = \_newsRepository.GetNews();

}

return \_publishers;

}

public void SaveChanged()

{

\_newsRepository.Save(\_publishers);

}

public void RemovePublisher(string publisherName)

{

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

SaveChanged();

}

public void RemoveCategory(string publisherName, string categoryName)

{

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

if (publisher == null) return;

publisher.RemoveCategory(categoryName);

SaveChanged();

}

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;

}

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace bttuan3

{

public partial class AddFeedForm : Form

{

public readonly NewsFeedManager \_newManager;

public bool HasChanges { get; set; }

public AddFeedForm(NewsFeedManager newManager)

{

\_newManager = newManager;

InitializeComponent();

}

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

{

var publishers = \_newManager.GetNewsFeed();

foreach(var p in publishers)

{

cbbPublishers.Items.Add(p.Name);

}

}

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

{

var pName = cbbPublishers.Text;

var cName = txtCategoryName.Text;

var rssl = txtRssLink.Text;

if(string.IsNullOrWhiteSpace(pName) ||

string.IsNullOrWhiteSpace(cName) ||

string.IsNullOrWhiteSpace(rssl))

{

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

return;

}

HasChanges = true;

var success = \_newManager.AddCategory(pName, cName, rssl, false);

if (success)

{

ClearForm();

return;

}

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

{

\_newManager.AddCategory(pName, cName, rssl, true);

}

ClearForm();

}

private void ClearForm()

{

cbbPublishers.Text = "";

txtCategoryName.Text = "";

txtRssLink.Text = "";

}

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

{

this.Close();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net;

using System.Text;

using System.Threading.Tasks;

namespace bttuan3.RSSFeed

{

public class RssReader

{

private readonly NewsParser \_parser;

public RssReader(NewsParser 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);

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

using System.Xml;

namespace bttuan3.RSSFeed

{

public class NewsParser

{

public List<Article> ParseXml(string xmlContent)

{

var document = new XmlDocument();

document.LoadXml(xmlContent);

var articles = new List<Article>();

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

foreach(XmlNode node in itemNode)

{

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 DateTime ParseDate(string innerText)

{

try

{

return DateTime.Parse(innerText);

}

catch

{

return DateTime.Now;

}

}

private string StripHtml(string content)

{

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

}

}

}