МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний аерокосмічний університет ім. М.Є. Жуковського “ХАІ”

Кафедра 603

Лабораторна робота № 2

з дисципліни «Реінжиніринг програмного забезпечення»

Розробка WEB-додатків з REST архітектурою.

ХАІ.603.657ІПЗ.15О.05010302.116301.ІПЗ

Виконав студент: гр. 657ІПЗ Апанасенко Ю.В.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (№ групи) (П.І.Б)

(підпис, дата)

Перевірив: доц., Розсоха С.В.

(наукова ступінь, вчене звання, П.І.Б)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(бал, підпис, дата)

2015

Цель: изучение средств методов реализации REST архитектуры для взаимодействия между клиентом и сервером WEB-приложений.

**Постановка задачи**

Разработать WEB-приложение “Заметки”. Для реализации приложения использовать шаблон проектирования MVC в любой из популярных платформ. Архитектуру приложения необходимо построить таким образом чтобы основные функции работы приложения выполнялись через HTTP запрос. При этом для каждой функции запрос должен быть свой.

Приложение должно выполнять следующие функции:

1. Добавление заметок.
2. Удаление заметок.
3. Пометка заметки как выполненной.
4. Отображение заметок по критериям:
5. выполненные заметки
6. не выполненные заметки
7. все заметки

**Ход работы**

В качестве платформы для реализации приложения была выбрана .NET 4.5. Так же был выбран ASP NET MVC 4 в котором уже реализован MVC для разработки web-приложений. Для реализации view приложения использовались серверные страницы cshtml. В качестве тестового web-сервера использовался IIS. Разработка велась в Visual Studio 2012.

В ходе реализации приложения были разработаны такие классы:

* ToDoModel – модель заметки
* ToDoController - все запросы и первичная обработка ответов происходит через этот класс.
* ToDoRepository – класс для хранения и получения состояния объектов.

Для отображения формы была создана страница evernote.jsp. На рисунке 1 изображена UML-диаграмма разработанного приложения.



Рисунок 1 - UML-диаграмма приложения Заметки.

Листинги файлов приложения приведены ниже.

Листинги файлов приложения приведены ниже.

Листинг файла ToDoRepository.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using laba1MVC.Models;

namespace laba1MVC.Repository

{

public class List

{

public List<ToDoModel> list;

}

public static class ToDoRepository

{

public static void Save(List<ToDoModel> list)

{

var writer = new System.Xml.Serialization.XmlSerializer(typeof(List<ToDoModel>));

string path = Environment.GetFolderPath(Environment.SpecialFolder.MyDocuments) + "//SerializationOverview.xml";

var wfile = new System.IO.StreamWriter(path);

writer.Serialize(wfile, list);

wfile.Close();

}

public static List<ToDoModel> Get()

{

string path = Environment.GetFolderPath(Environment.SpecialFolder.MyDocuments) + "//SerializationOverview.xml";

System.Xml.Serialization.XmlSerializer reader =

new System.Xml.Serialization.XmlSerializer(typeof(List<ToDoModel>));

System.IO.StreamReader file = new System.IO.StreamReader(

path);

List<ToDoModel> overview = (List<ToDoModel>)reader.Deserialize(file);

file.Close();

return overview;

}

}

}

Листинг файла RestApiController.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Http;

using System.Web.Mvc;

using laba1MVC.Models;

using laba1MVC.Repository;

namespace laba1MVC.Controllers

{

public class RestApiController : ApiController

{

//

// GET: /RestApi/

public IEnumerable<ToDoModel> GetModels()

{

return ToDoRepository.Get();

}

}

}

Листинг файла ToDoController.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using laba1MVC.Models;

using laba1MVC.Repository;

namespace laba1MVC.Controllers

{

public class ToDoController : Controller

{

//

// GET: /ToDo/

List<ToDoModel> ToDoList = new List<ToDoModel>();

[HttpGet]

// [OutputCacheAttribute(VaryByParam = "\*", Duration = 1, NoStore = true)]

public ActionResult Index()

{

List<ToDoModel> temp = ToDoRepository.Get();

if (temp != null)

ToDoList = temp;

ToDoRepository.Save(ToDoList);

return View(ToDoList);

}

[HttpGet]

public ActionResult Save()

{

return View();

}

[HttpPost]

public ActionResult Save(String toDo)

{

ToDoList = ToDoRepository.Get();

ToDoModel newModel = new ToDoModel();

newModel.ToDo = toDo;

newModel.State = false;

ToDoList.Add(newModel);

ToDoRepository.Save(ToDoList);

return RedirectToAction("Index") ;

}

[HttpPost]

public ActionResult SetChecked(string toDo, bool value)

{

ToDoList = ToDoRepository.Get();

ToDoModel editModel = ToDoList.First(x => x.ToDo == toDo);

editModel.State = value;

ToDoRepository.Save(ToDoList);

//return Redirect(Request.UrlReferrer.ToString());

return RedirectToAction("Index", "ToDo");

}

[HttpGet]

public ActionResult Delete(string toDo)

{

ToDoList = ToDoRepository.Get();

ToDoModel model = ToDoList.Find(x => x.ToDo == toDo);

ToDoList.Remove(model);

ToDoRepository.Save(ToDoList);

return RedirectToAction("Index");

}

[HttpGet]

public ActionResult ClearCompleted()

{

ToDoList = ToDoRepository.Get();

List<ToDoModel> models = ToDoList.FindAll(x => x.State == true);

ToDoList.RemoveAll(x => x.State == true);

ToDoRepository.Save(ToDoList);

return RedirectToAction("Index");

}

[HttpGet]

public ActionResult Sort(string condition)

{

ToDoList = ToDoRepository.Get();

List<ToDoModel> result = new List<ToDoModel>();

switch (condition)

{

case "All":

result = ToDoList;

break;

case "Active":

result = ToDoList.FindAll(x => x.State == false);

break;

case "Completed":

result = ToDoList.FindAll(x => x.State == true);

break;

}

ToDoRepository.Save(ToDoList);

return View("Index", result);

}

void Session\_Start(object sender, EventArgs e)

{

}

void Session\_End(object sender, EventArgs e)

{

}

}

}

Листинг файла ToDoModel.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace laba1MVC.Models

{

public class ToDoModel

{

public String ToDo { get; set; }

public bool State { get; set; }

}

}

Листинг файла RouteConfig.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Http;

using System.Web.Mvc;

using System.Web.Routing;

namespace laba1MVC

{

public class RouteConfig

{

public static void RegisterRoutes(RouteCollection routes)

{

routes.IgnoreRoute("{resource}.axd/{\*pathInfo}");

routes.MapHttpRoute(

name: "DefaultApi",

routeTemplate: "api/{controller}/{id}",

defaults: new { id = RouteParameter.Optional }

);

routes.MapRoute(

name: "Default",

url: "{controller}/{action}/{id}",

defaults: new { controller = "ToDo", action = "Index", id = UrlParameter.Optional }

);

}

}

}

Листинг файла Index.cshtml

@model ICollection<laba1MVC.Models.ToDoModel>

<script>

function handleClick(cb) {

$.ajax({

url:"@Url.Action("SetChecked","ToDo")",

type: 'POST',

contentType: 'application/json;',

data: JSON.stringify({ toDo: cb.name, value: cb.checked}),

success: function (valid) {

if (valid) {

location.reload(true);

} else {

//show that id is not valid

}

}

});

}

</script>

<h2>Todos</h2>

<body>

<div id="content"></div>

@using (Html.BeginForm("Save","ToDo",FormMethod.Post))

{

@Html.TextBox("ToDo");

<input type="submit" value="Add" />

}

<br />

@Html.ActionLink("Clear complited", "ClearCompleted", "ToDo")

@Html.ActionLink("All", "Sort", "ToDo", new {condition = "All" }, null)

@Html.ActionLink("Active", "Sort", "ToDo", new {condition = "Active" }, null)

@Html.ActionLink("Completed", "Sort", "ToDo", new {condition = "Completed" }, null)

</body>

@section scripts {

<script>

var apiCustomers = 'api/RestApi';

$(document).ready(function () {

// Send an AJAX request

var rows = '';

// $('#content').empty();

$.getJSON(apiCustomers,

function (data) {

$.each(data, function (key, val) {

var state ='';

if (val.State == true)

state = 'checked';

else

state = '';

var url = '/ToDo/Delete?ToDo=' + val.ToDo;

rows = rows +

'<input type="checkbox" name = "' + val.ToDo + '" onclick = "handleClick(this)" ' + state + ' />' + val.ToDo

+' <a href="'+url+'">Delete</a>' + '<br>';

});

$("#content").append(rows);

});

});

</script>

}

Результат работы приложения представлен на рисунке 2.

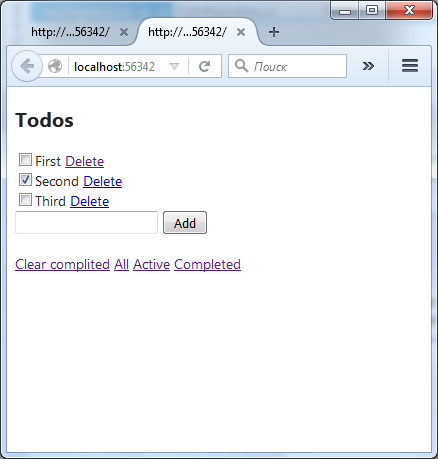


Рисунок 2 - Результат работы приложения.

Вывод: в ходе выполнения лабораторной работы с использованием ASP .NET MVC 4 было разработано web-приложение заметки. Обмен данными в приложении происходит через HTTP запросы для каждой отдельной функции. Разработка приложения позволила получить навыки использования ASP .NET MVC 4 фреймворка для разработки подобного рода web-приложений.