|  |
| --- |
| **BAHRIA UNIVERSITY (KARACHI CAMPUS)**    Open Ended Lab 1 – SPRING SEMESTER – 2020    **(Software Design and Architecture - SEL457)**    Class: **BSE 4A** (Morning)  Course Instructor: **Engr Muhammad Rehan Baig**  Max Marks: 15    Student’s Name: Shafaq Fatima Amir Reg. No: 57235      **Note:** Folder structure must be in given format  Name\_Enrollment\_Registration **[CLO 4]** |

**SCENERIO:**

USA based marketing agency want to develop web based application for automate their marketing strategies. An application contains following modules.

* Registration/Authorization(Signup/Login)
*  Scheduled based Ads posting on defined websites.
* Content Management System.
* Accounting Module for Ads that hourly calculate and charge clients.
* REST Api’s for other devices to connect.

**Q1 Design** Data flow Diagram, Class Diagram and ERD diagram for the given Scenario.

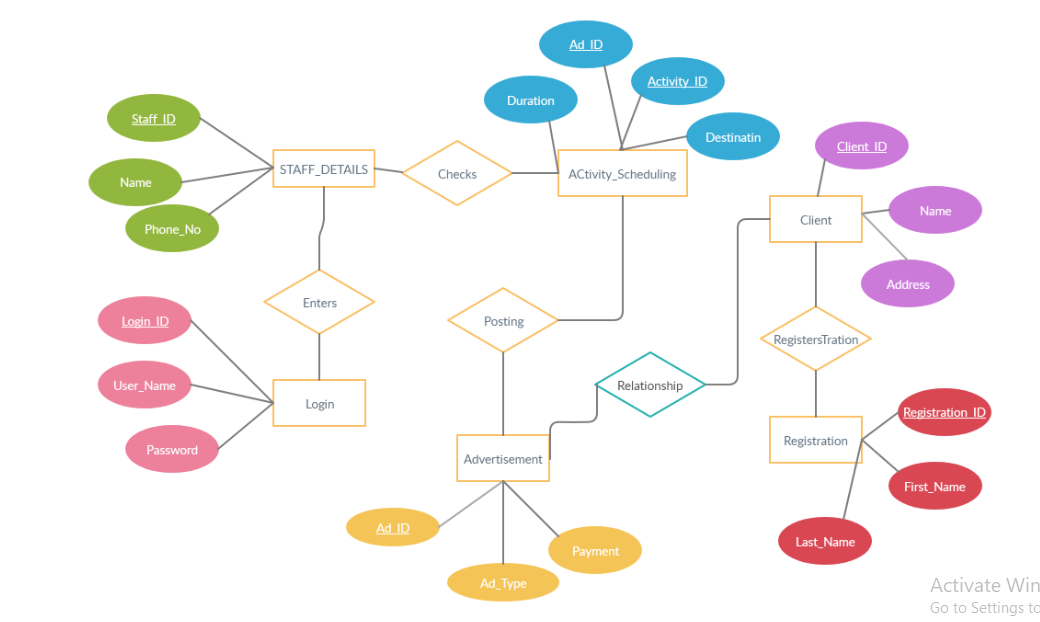
**[7 marks]**

**Answer:**









**Q2 Identify** and **Implement** best suited design patterns for the given scenario (**Note**: Multiple

Design patterns can be implemented) **[8 marks]**

**Answer:**

**Singleton pattern for Login**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LoginModule

{

public class StaffAction

{

private static StaffAction \_obj;

public static Staff s;

private StaffAction(StaffLoginId staff) {

s = new Staff();

s.Name = staff.Username;

}

static public StaffAction Logged(StaffLoginId sta)

{

if (\_obj == null)

{

\_obj = new StaffAction(sta);

}

return \_obj;

}

public string getUsername() {

return s.Name ;

}

}

}

namespace LoginModule

{

public class Staff

{

public String StaffId { get; set; }

public String Name { get; set; }

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LoginModule

{

public class StaffLoginId

{

public string id { get; set; }

public string LoginId { get; set; }

public string Username{ get; set; }

public string password { get; set; }

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LoginModule

{

class Program

{

static void Main(string[] args)

{

StaffLoginId sl = new StaffLoginId();

sl.Username = "sana";

Staff s = new Staff();

s.Name = "sana";

StaffAction sa = StaffAction.Logged(sl);

Console.WriteLine(sa.getUsername());

Console.ReadLine();

}

}

}

**Chain of responsibility and iterator for RESTAPI**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ScheduleAdPosting

{

public class Advertisement

{

public int point;

public string name;

public Advertisement(int a, string name) {

this.point = a;

this.name = name;

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ScheduleAdPosting

{

public abstract class Approver

{

protected Approver Sucessor;

public void SetSucessor(Approver name)

{

this.Sucessor = name;

}

public abstract int AdCheck(Advertisement Ad);

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ScheduleAdPosting

{

public class HeadOfMarkting:Approver

{

public override int AdCheck(Advertisement Ad)

{

if (Ad.point == 100)

{

Console.WriteLine(Ad.point);

Console.WriteLine("Being Approved..");

}

else

{

Console.WriteLine("Not approved.");

return 0;

}

return 1;

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ScheduleAdPosting

{

public class Managr:Approver

{

public override int AdCheck(Advertisement Ad)

{

if (Ad.point == 0)

{

Console.WriteLine(Ad.point);

Console.WriteLine("Being Approved..");

}

else if (Sucessor != null)

{

Sucessor.AdCheck(Ad);

}

return 1;

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ScheduleAdPosting

{

public class SeniorManage:Approver

{

public override int AdCheck(Advertisement Ad)

{

if (Ad.point == 50)

{

Console.WriteLine(Ad.point);

Console.WriteLine("Being Approved..");

}

else if (Sucessor != null)

{

Sucessor.AdCheck(Ad);

}

return 1;

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ScheduleAdPosting

{

public abstract class Aggregate

{

public abstract WebsiteIterator CreateWebsiteIterator();

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ScheduleAdPosting

{

public class ConcereteIterator : WebsiteIterator

{

private ConcreteAggregate \_aggregate;

private int \_current = 0;

public ConcereteIterator(ConcreteAggregate aggregate)

{

this.\_aggregate = aggregate;

}

public override object First()

{

return \_aggregate[0];

}

public override object CurrentItem()

{

return \_aggregate[\_current];

}

public override bool IsDone()

{

return \_current >= \_aggregate.Count;

}

public override object Next()

{

object ret = null;

if (\_current < \_aggregate.Count - 1)

{

ret = \_aggregate[++\_current];

}

return ret;

}

}

}

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ScheduleAdPosting

{

public class ConcreteAggregate : Aggregate

{

private ArrayList \_items = new ArrayList();

public override WebsiteIterator CreateWebsiteIterator()

{

return new ConcereteIterator(this);

}

public int Count

{

get { return \_items.Count; }

}

public object this[int index]

{

get { return \_items[index]; }

set { \_items.Insert(index, value); }

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ScheduleAdPosting

{

public abstract class WebsiteIterator

{

public abstract object First();

public abstract object Next();

public abstract bool IsDone();

public abstract object CurrentItem();

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ScheduleAdPosting

{

class Program

{

static void Main(string[] args)

{

int approved;

Approver SM = new SeniorManage();

Approver M = new Managr();

Approver HOM = new HeadOfMarkting();

M.SetSucessor(SM);

SM.SetSucessor(HOM);

Advertisement AD1 = new Advertisement(50, "K&Ns");

approved= SM.AdCheck(AD1);

Console.WriteLine();

ConcreteAggregate a = new ConcreteAggregate();

if (approved ==1)

{

a[0] = AD1.name;

}

WebsiteIterator i = a.CreateWebsiteIterator();

Console.WriteLine("Iterating over ADS collection:");

object item = i.First();

while (item != null)

{

Console.WriteLine(item);

item = i.Next();

}

Console.ReadKey();

}

}

}

**Façade for CMS**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Content\_Management\_System

{

public class MainMenuOption

{

private SchedulingBasedAdPosting a;

private ClientCharges b;

private StatusCheck c;

public MainMenuOption() {

a = new SchedulingBasedAdPosting();

b = new ClientCharges();

c = new StatusCheck();

}

public void ClientChargesAndScheduling() {

Console.WriteLine(a.SchedulingInfo());

Console.WriteLine(b.ClientChargesAd() );

}

public void SC() {

Console.WriteLine(c.Status());

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Content\_Management\_System

{

public class StatusCheck

{

public string Status()

{

return "status";

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Content\_Management\_System

{

public class SchedulingBasedAdPosting

{

public string SchedulingInfo() {

return "Schedule Info";

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Content\_Management\_System

{

public class ClientCharges

{

public string ClientChargesAd() {

return "Client Charges";

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Content\_Management\_System

{

class Program

{

static void Main(string[] args)

{

MainMenuOption mm = new MainMenuOption();

mm.ClientChargesAndScheduling();

mm.SC();

Console.ReadLine();

}

}

}

**Strategy Pattern for Client Charges**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ClientCharges

{

public class ClientBased

{

private IStrategy \_strategy;

public ClientBased()

{ }

public ClientBased(IStrategy strategy)

{

this.\_strategy = strategy;

}

public void SetStrategy(IStrategy strategy)

{

this.\_strategy = strategy;

}

public interface IStrategy

{

object DoAlgorithm(int data);

}

public double AdCharges(double charge,int hours)

{

Console.WriteLine();

Console.WriteLine("\nHours : "+ hours);

Console.WriteLine("Client Charges : " + charge);

double result = charge \* hours;

Console.Write("Total Charge = "+ result + "\n");

Console.WriteLine();

return result;

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ClientCharges

{

class Program

{

static void Main(string[] args)

{

ClientBased KandN = new ClientBased();

KandN.AdCharges(90,4);

ClientBased Samsung = new ClientBased();

Samsung.AdCharges(50,5);

Console.WriteLine();

Console.ReadLine();

}

}

}

**Adapter pattern for RESTAPI**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace RESTApiAdapter

{

public class TargetDeviceInterface

{

public virtual void Request()

{

Console.WriteLine("Calling targetted request");

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace RESTApiAdapter

{

public class Adapter: TargetDeviceInterface

{

private RESTAPIAdaptee \_adaptee = new RESTAPIAdaptee();

public override void Request()

{

\_adaptee.SpecificRequest();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace RESTApiAdapter

{

public class RESTAPIAdaptee

{

public void SpecificRequest()

{

Console.WriteLine("Connecting with the device");

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace RESTApiAdapter

{

class Program

{

static void Main(string[] args)

{

//REQUESTING CONNECTION

TargetDeviceInterface target = new Adapter();

target.Request();

Console.ReadKey();

}

}

}

Project Link

https://github.com/safame108/OEL1.git