G4 GUIDANCE

Submitted to:

Ma'am Amna Mirza

Submitted by:

Ahmad Sarwar

(BSEF19M034)

Subject:

Object Oriented Analysis and Design

Sequence of Implementation of Classes from UMLs

Classes with having no dependency:

1- Management
2- Student
3- Blog
4- Playlist
5- Degree
6- Entrance Test

Classes with some dependency:

1- Admin (Depends upon Author, University)
2- Author (Depends upon Blog, Playlist)
3- Department (Depends upon Degree)
4- University (Depends upon Department)

Management (Interface)

```
 class management
{
    public virtual void add()
    {
        //abstract function nothing here
    }
    public virtual object Read()
    {
        //abstract function nothing here
        return null;
    }
    public virtual void update()
    {
        //abstract function nothing here
    }
    public virtual void Delete()
    {
        //abstract function nothing here
    }
}
```

Student

```
class Student
{
    private int Id;
    private string name;
    private string username;
    private string phoneno;
    private string addess;
    private int postelcode;
    private string cardno;
    private int csv;
    private bool status;

    public Student()//default constrcutor
    {
        Id = -1;
        name = string.Empty;
        username = string.Empty;
        phoneno = string.Empty;
        addess = string.Empty;
        postelcode = -1;
        cardno = string.Empty;
        csv = -1;
        status = false;
    }
    public Student(int id, string Name, string userName, string
     Phone, string Adress, int Postelcode, string CardNO, int
     Csv, bool Status)  //parametrized constructor
    {
        Id = id;
        name = Name;
        username = userName;
        phoneno = Phone;
        addess = Adress;
        postelcode = Postelcode;
        cardno = CardNO;
        csv = Csv;
        status = Status;
    }
    public bool Status
    {
        get { return status; }
        set { status = value; }
    }

    public int CSV
    {
        get { return csv; }
        set { csv = value; }
    }

    public string CardNo
    {
        get { return cardno; }
        set { cardno = value; }
    }

    public int PostelCode
    {
        get { return postelcode; }
        set { postelcode = value; }
    }

    public string Address
    {
        get { return addess; }
        set { addess = value; }
    }

    public string PhoneNo
    {
        get { return phoneno; }
        set { phoneno = value; }
    }

    public int ID
    {
        get { return Id; }
        set { Id = value; }
    }
    public string Name
    {
        get { return name; }
        set { name = value; }
    }
    public string Username
    {
        get { return username; }
        set { username = value; }
    }
}
```

Blog

```
    class Blog
    {
        private int Id;
        private string title;
        private string category;
        private string date;
        private string description;
        private string authorname;
        public Blog()
        {
            Id = -1;
            title = string.Empty;
            category = string.Empty;
            date = string.Empty;
            description = string.Empty;
            authorname = string.Empty;
        }
        public Blog(int id, string t_title, string cat, string DATE,
string descp, string author)
        {
            Id = id;
            title = t_title;
            category = cat;
            date = DATE;
            description = descp;
            authorname = author;
        }
        public string AuthorName
        {
            get { return authorname; }
            set { authorname = value; }
        }

        public string Description
        {
            get { return description; }
            set { description = value; }
        }

        public string Date
        {
            get { return date; }
            set { date = value; }
        }

        public string Category
        {
            get { return category; }
            set { category = value; }
        }

        public string Title
        {
            get { return title; }
            set { title = value; }
        }

        public int ID
        {
            get { return Id; }
            set { Id = value; }
        }

    }
```

Playlist

```
    class Playlist
    {
        private int Id;
        private string title;
        private string category;
        private string date;
        private string description;
        private string video;

        public Playlist()
        {
            Id = -1;
            title = string.Empty;
            category = string.Empty;
            date = string.Empty;
            description = string.Empty;
            video = string.Empty;
        }
        public Playlist(int id, string t_title, string cat, string
         DATE, string descp, string Video_Link)
        {
            Id = id;
            title = t_title;
            category = cat;
            date = DATE;
            description = descp;
            video = Video_Link;
        }
        public string VideoLink
        {
            get { return video; }
            set { video = value; }
        }

        public string Description
        {
            get { return description; }
            set { description = value; }
        }

        public string Date
        {
            get { return date; }
            set { date = value; }
        }

        public string Category
        {
            get { return category; }
            set { category = value; }
        }

        public string Title
        {
            get { return title; }
            set { title = value; }
        }

        public int ID
        {
            get { return Id; }
            set { Id = value; }
        }

    }
```

Degree

```
    class Degree
    {
        private int Id;
        private string name;
        private double merit;
        private string forumula;
        private int duration;

        public Degree()
        {
            Id = -1;
            name = string.Empty;
            merit = 0.000;
            forumula = string.Empty;
            duration = -1;
        }
        public Degree(int ID, string NAME, double MERIT, string
form,        int dur)
        {
            Id = ID;
            name = NAME;
            merit = MERIT;
            forumula = form;
            duration = dur;
        }
        public int Duration
        {
            get { return duration; }
            set { duration = value; }
        }

        public string Formula
        {
            get { return forumula; }
            set { forumula = value; }
        }

        public double Merit
        {
            get { return merit; }
            set { merit = value; }
        }

        public string Name
        {
            get { return name; }
            set { name = value; }
        }

        public int ID
        {
            get { return Id; }
            set { Id = value; }
        }

    }
```

Entrance Test

```
    class EnteranceTest
    {
        private int Id;
        private string name;
        private int totalmarks;
        private double marks;

        public EnteranceTest()
        {
            Id = -1;
            name = string.Empty;
            totalmarks = -1;
            marks = 0.000;
        }
        public EnteranceTest(int ID, string NAME, int TOTAL, double
MARKS)
        {
            Id = ID;
            name = NAME;
            totalmarks = TOTAL;
            marks = MARKS;
        }
        public double Marks
        {
            get { return marks; }
            set { marks = value; }
        }

        public int Totalmarks
        {
            get { return totalmarks; }
        }

        public string Name
        {
            get { return name; }
            set { name = value; }
        }

        public int ID
        {
            get { return Id; }
            set { Id = value; }
        }

    }
```

Department

```
    class Department
    {
        private int Id;
        private string name;
        private List<Degree> deg = new List<Degree>();
        public string NAME
        {
            get { return name; }
            set { name = value; }
        }

        public int ID
        {
            get { return Id; }
            set { Id = value; }
        }
        public void addDegree(int id, string name, double merit,
string formula, int duration)
        {
            Degree d = new Degree(id, name, merit, formula,
duration);
            deg.Add(d);
        }
        public Degree getDegree(int id)
        {
            foreach(Degree D in deg)
            {
                if(D.ID==id)
                {
                    return D;
                }
            }
            return null;
        }
        public void updateDegree(int id, string name, double merit,
string formula, int duration)
        {
            foreach (Degree D in deg)
            {
                if (D.ID == id)
                {
                    D.Name = name;
                    D.Merit = merit;
                    D.Formula = formula;
                    D.Duration = duration;
                }
            }
        }
        public void deleteDegree(int id)
        {
            foreach (Degree D in deg)
            {
                if (D.ID == id)
                {
                    deg.Remove(D);
                }
            }
        }
    }
```

University

```
class University
{
    private int Id;
    private string name;
    private string type;
    private string description;
    private List<Department> deg = new List<Department>();
    public string Description
    {
        get { return description; }
        set { description = value; }
    }

    public string TYPE
    {
        get { return type; }
        set { type = value; }
    }

    public string NAME
    {
        get { return name; }
        set { name = value; }
    }

    public int ID
    {
        get { return Id; }
        set { Id = value; }
    }

    public void addDepartment(int id, string name)
    {
        Department d = new Department();
        d.ID = id;
        d.NAME = name;
        deg.Add(d);
    }
    public Department getDegree(int id)
    {
        foreach (Department D in deg)
        {
            if (D.ID == id)
            {
                return D;
            }
        }
        return null;
    }
    public void updateDegree(int id, string name)
    {
        foreach (Department D in deg)
        {
            if (D.ID == id)
            {
                D.NAME = name;
            }
        }
    }
    public void deleteDegree(int id)
    {
        foreach (Department D in deg)
        {
            if (D.ID == id)
            {
                deg.Remove(D);
            }
        }
    }
    public dynamic manageDegree(params dynamic[] arg)
    {
        Department dep=null;
        foreach(Department d in deg)
        {
            if(d.ID==arg[2])
            {
                dep = d;
                break;
            }
        }
        if(arg[1]==1)
        {
            dep.addDegree(arg[1], arg[2], arg[3], arg[4],
        arg[5]);
        }
        if (arg[1] == 2)
```

```
            {
                dep.deleteDegree(arg[1]);
            }
            if (arg[1] == 1)
            {
                return dep.getDegree(arg[1]);
            }
            if (arg[1] == 1)
            {
                dep.updateDegree(arg[1], arg[2], arg[3], arg[4],
            arg[5]);
            }
            return null;
        }
    }
```

Author

```
    class Author : management
    {
        private int Id;
        private string name;
        private string username;
        private string email;

        public string Email
        {
            get { return email; }
            set { email = value; }
        }

        public string UserName
        {
            get { return username; }
            set { username = value; }
        }

        public string Name
        {
            get { return name; }
            set { name = value; }
        }
        public int ID
        {
            get { return Id; }
            set { Id = value; }
        }
        //All the below given fuctions will be called from HTML
routes
        public void add(Blog a)
        {
            string conString = @"Data
Source=(localdb)\ProjectsV13;Initial Catalog=G4Guidacne;Integrated
Security=True;Connect
Timeout=30;Encrypt=False;TrustServerCertificate=False;ApplicationInt
ent=ReadWrite;MultiSubnetFailover=False";
            SqlConnection con = new SqlConnection(conString);
            con.Open();
            string query = $"Insert into Blog
values('{a.ID}','{a.Title}',{a.Category},'{a.AuthorName}','{a.Descri
ption}','{a.Date}'";
            SqlCommand cmd = new SqlCommand(query, con);
            int i = cmd.ExecuteNonQuery();
            con.Close();
        }
        public void update(Blog a)
        {
            string conString = @"Data
Source=(localdb)\ProjectsV13;Initial Catalog=G4Guidacne;Integrated
Security=True;Connect
Timeout=30;Encrypt=False;TrustServerCertificate=False;ApplicationInt
ent=ReadWrite;MultiSubnetFailover=False";
            SqlConnection con = new SqlConnection(conString);
            con.Open();
            string query = $"Update Blog set
values('{a.ID}','{a.Title}',{a.Category},'{a.AuthorName}','{a.Descri
ption}','{a.Date}' where id='{a.ID}'";
            SqlCommand cmd = new SqlCommand(query, con);
            int i = cmd.ExecuteNonQuery();
            con.Close();
        }
        public void Delete(Blog a)
        {
            string conString = @"Data
Source=(localdb)\ProjectsV13;Initial Catalog=G4Guidance;Integrated
Security=True;Connect
Timeout=30;Encrypt=False;TrustServerCertificate=False;ApplicationInt
ent=ReadWrite;MultiSubnetFailover=False";
            SqlConnection con = new SqlConnection(conString);
            con.Open();
            string query = $"delete from Blog where Id  = {a.ID}";
            SqlCommand cmd = new SqlCommand(query, con);
            int i = cmd.ExecuteNonQuery();
        }
        public object Read(Blog a)
        {
            string conString = @"Data
Source=(localdb)\ProjectsV13;Initial Catalog=G4Gudiance;Integrated
Security=True;Connect
Timeout=30;Encrypt=False;TrustServerCertificate=False;ApplicationInt
ent=ReadWrite;MultiSubnetFailover=False";
            SqlConnection con = new SqlConnection(conString);
            con.Open();
            string query = $"select * from Blog";
            SqlCommand cmd = new SqlCommand(query, con);
            SqlDataReader sdr = cmd.ExecuteReader();
            object blog=Deserlize(sdr);
            return blog;
        }
    }
```

Admin

```
class admin : management
    {
        private int Id;
        private string name;
        private string username;
        private string email;

        public string Email
        {
            get { return email; }
            set { email = value; }
        }

        public string UserName
        {
            get { return username; }
            set { username = value; }
        }

        public string Name
        {
            get { return name; }
            set { name = value; }
        }

        public int ID
        {
            get { return Id; }
            set { Id = value; }
        }
        //All the below given fuctions will be called from HTML
routes
        public void add(Author a)
        {
            string conString = @"Data
Source=(localdb)\ProjectsV13;Initial Catalog=G4Guidacne;Integrated
Security=True;Connect
Timeout=30;Encrypt=False;TrustServerCertificate=False;ApplicationInt
ent=ReadWrite;MultiSubnetFailover=False";
            SqlConnection con = new SqlConnection(conString);
            con.Open();
            string query = $"Insert into author
values('{a.ID}','{a.Name}',{a.UserName},'{a.Email}'";
            SqlCommand cmd = new SqlCommand(query, con);
            int i = cmd.ExecuteNonQuery();
            con.Close();
        }
        public void update(Author a)
        {
            string conString = @"Data
Source=(localdb)\ProjectsV13;Initial Catalog=G4Guidacne;Integrated
Security=True;Connect
Timeout=30;Encrypt=False;TrustServerCertificate=False;ApplicationInt
ent=ReadWrite;MultiSubnetFailover=False";
            SqlConnection con = new SqlConnection(conString);
            con.Open();
            string query = $"Update author set
values('{a.ID}','{a.Name}',{a.UserName},'{a.Email}' where
id='{a.ID}'";
            SqlCommand cmd = new SqlCommand(query, con);
            int i = cmd.ExecuteNonQuery();
            con.Close();
        }
        public void Delete(Author a)
        {
            string conString = @"Data
Source=(localdb)\ProjectsV13;Initial Catalog=g4Guidacne;Integrated
Security=True;Connect
Timeout=30;Encrypt=False;TrustServerCertificate=False;ApplicationInt
ent=ReadWrite;MultiSubnetFailover=False";
            SqlConnection con = new SqlConnection(conString);
            con.Open();
            string query = $"delete from author where Id  = {a.ID}";
            SqlCommand cmd = new SqlCommand(query, con);
            int i = cmd.ExecuteNonQuery();
        }
        public object Read(Author a)
        {
            string conString = @"Data
Source=(localdb)\ProjectsV13;Initial Catalog=G4Guidacne;Integrated
Security=True;Connect
Timeout=30;Encrypt=False;TrustServerCertificate=False;ApplicationInt
ent=ReadWrite;MultiSubnetFailover=False";
            SqlConnection con = new SqlConnection(conString);
            con.Open();
            string query = $"select * from author";
            SqlCommand cmd = new SqlCommand(query, con);
            SqlDataReader sdr = cmd.ExecuteReader();
            object author = Deserlize(sdr);
            return author;
        }
    }
```

Exceptions List

I. AccessViolationException
II. ArgumentNullException
III. ArgumentOutOfRangeException
IV. DivideByZeroException
V. MissingMemberException
VI. NullReferenceException
VII. IndexOutOfRangeException
VIII. TypeAccessException
IX. DuplicateWaitObjectException
X. InsufficientMemoryException
XI. SQLClientInfoException
XII. InvalidObjectNameException
XIII. objectDisposedException
XIV. unauthorizedaccessexception