

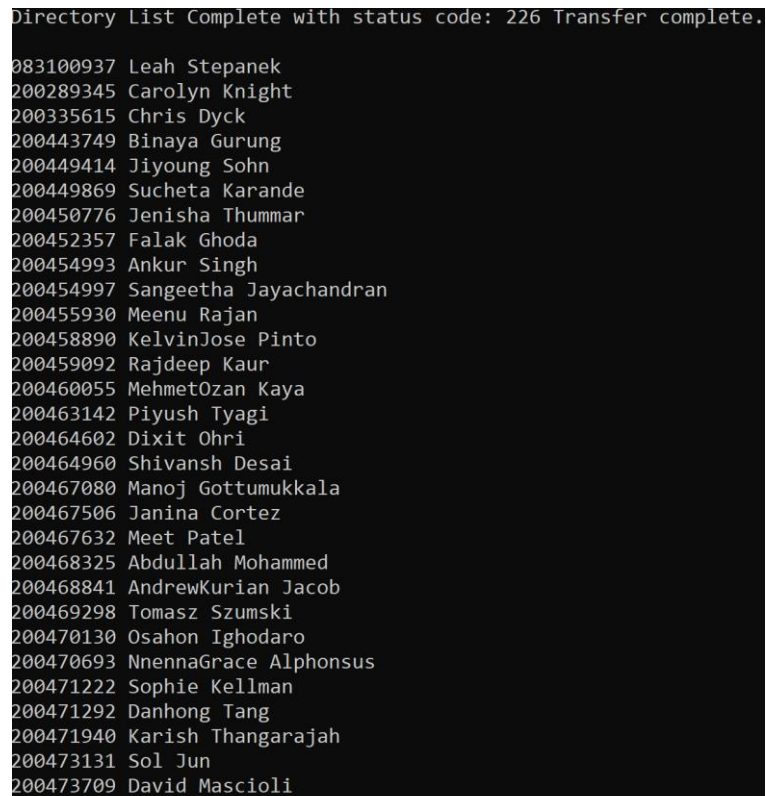
# ASSIGNMENT- 3

## 1.Retrieve a list of directories from the FTP

### 1.Output the directories to the Console

```
List<string> directories = FTP.GetDirectory(Constants.FTP.BaseUrl);
foreach (var directory in directories)
{
    Console.WriteLine(directory);
}
```

Explanation: The first statement creates a directories list. GetDirectory method returns the names of the directories and stores them in the directories list. The foreach loop iterates through each element in the directories list and prints them. The output to the console looks like the below image.



```
Directory List Complete with status code: 226 Transfer complete.
2003100937 Leah Stepanek
200289345 Carolyn Knight
200335615 Chris Dyck
200443749 Binaya Gurung
200449414 Jiyoungh Sohn
200449869 Sucheta Karande
200450776 Jenisha Thummar
200452357 Falak Ghoda
200454993 Ankur Singh
200454997 Sangeetha Jayachandran
200455930 Meenu Rajan
200458890 KelvinJose Pinto
200459092 Rajdeep Kaur
200460055 MehmetOzan Kaya
200463142 Piyush Tyagi
200464602 Dixit Ohri
200464960 Shivansh Desai
200467080 Manoj Gottumukkala
200467506 Janina Cortez
200467632 Meet Patel
200468325 Abdullah Mohammed
200468841 AndrewKurian Jacob
200469298 Tomasz Szumski
200470130 Osahon Ighodaro
200470693 NnennaGrace Alphonsus
200471222 Sophie Kellman
200471292 Danhong Tang
200471940 Karish Thangarajah
200473131 Sol Jun
200473709 David Mascioli
```

## 2. Extract the data from your directory

1. In your directory, you should be able to see the files named **info.csv** and **myimage.jpg**
2. Report on the files in the directory

```

static void Main(string[] args)
{
    List<string> directories = FTP.GetDirectory(Constants.FTP.BaseUrl);
    List<Student> students = new List<Student>();
    var direct = "200467080 Manoj Gottumukkala";
    //foreach (var directory in directories)
    //{
    //    Console.WriteLine(directory);
    //}

    foreach (var directory in directories)
    {
        if(direct==directory)
        {
            Console.WriteLine("Directory exists");
            if(FTP.FileExists(Constants.FTP.BaseUrl+"/"+direct+"/info.csv"))
            {
                Console.WriteLine("info.csv file exists");
            }
            if(FTP.FileExists(Constants.FTP.BaseUrl + "/" + direct + "/myimage.jpg"))
            {
                Console.WriteLine("myimage.jpg exists");
            }
        }
    }
}

```

#### Explanation:

A directories list is created. Directories list has names of the student folders. When iterating through foreach loop, if the name of the directory(directory) equals to my folder name(direct), a message(Directory exists) is displayed on the Console. In the next few statements, we check if the files(info.csv,myimage.jpg) exists. The output to the console is

```

Directory exists
info.csv file exists
myimage.jpg exists

```

### 3. Extract the data from these files.

```

foreach (var directory in directories)
{
    if(direct==directory)
    {
        Console.WriteLine("Directory exists");
        if(FTP.FileExists(Constants.FTP.BaseUrl+"/"+direct+"/info.csv"))
        {
            Student student = new Student();
            Console.WriteLine("info.csv file exists");
            var fileBytes = FTP.DownloadFileBytes(Constants.FTP.BaseUrl + "/" + directory + "/info.csv");
            string infoCsvData = Encoding.UTF8.GetString(fileBytes, 0, fileBytes.Length);
            string[] lines = infoCsvData.Split("\r\n", StringSplitOptions.RemoveEmptyEntries);
            student.FromCsv(lines[1]);
            Console.WriteLine(student);
        }
        if (FTP.FileExists(Constants.FTP.BaseUrl + "/" + direct + "/myimage.jpg"))
        {
            Console.WriteLine("myimage.jpg exists");
        }
    }
}

```

### Explanation of the code:

- [illegible]

- The downloaded image file is converted to base64 format and displayed.

```

string desktopPath = Environment.GetFolderPath(Environment.SpecialFolder.Desktop);

//Build a image file path from the original image path
FileInfo fileinfo = new FileInfo($"{desktopPath}\\myimage.jpg ");

//Provide an Image from a file on your Desktop
Image image = Image.FromFile(fileinfo.FullName);

//Convert Image to Base64 encoded text
string base64image = Converter.ImageToBase64(image, ImageFormat.Jpeg);

//Output the Base64 encoded text
Console.WriteLine(base64image);

```

Saving the image data programmatically into the info.csv file.

```

string[] lines = { "StudentId,FirstName,LastName,DateOfBirth,ImageData", "200467080,Nanoj,Gottumukkala,18/10/97, /9j/4AAQSkZJRgABAQEAYABgAAD/4Tr+RXhpZgAATU0BAQAABgAAALAAIAAAAmAAITyGE5ANPMAAABAAEAAAEAAIAAAAmAAATIAEyAAIAAAAUAAATrodpaAQAAAAAB/

// Set a variable to the Documents path.
string docPath =
    Environment.GetFolderPath(Environment.SpecialFolder.Desktop);

using (StreamWriter outputFile = new StreamWriter(Path.Combine(docPath, "data.csv")))
{
    foreach (string line in lines)
        outputFile.WriteLine(line);
}

```

### 3.Add a new Model (class) that represents the details found in your directory.

a. Add Age as a calculated read-only property.

```

//Calculating age.
string currentyear = DateTime.Now.Year.ToString(); //Takes the current year.
try
{
    int x = int.Parse(currentyear); //Converts the string to int type.
    int p = DateOfBirthString.Length - 2; //Takes the index of the second element from the last.
    int y = int.Parse(DateOfBirthString.Substring(p)); //y takes only substrings For example in 18/10/1997 or 10/18/1997 only 97 is taken.
    if (y < 50) //If y<50, the user has given the input in the format 1997/10/18.
    {
        y = int.Parse(DateOfBirthString.Substring(2, 2)); //So, taking the 3th and 4th element.
    }
    int age = x - (1900 + y); //Adding 1900 to y and subtracting from 2020. i.e 2020-1997
}
catch
{
    //If the user doesn't enter the age, then by default 0 is set.
    age1.Add(0);
}

```

Explanation: The code is well explained with the comments.

**b. Add a ToString override to display to Student model as a string representation of its contents**

```
public override string ToString()
{
    return $"{StudentCode}-{FirstName},{LastName},{DateOfBirthString},{ImageData},{Record}";
}
```

Explanation: The ToString method returns the StudentCode, FirstName, LastName, DateOfBirthString, ImageData and Record created in the Student class.

**c. Add a FromDirectory method to extract data from the Directory name and fill the Student object**

```
public void FromDirectory(string direct)
{
    string[] directoryPart = direct.Split(" ", StringSplitOptions.None); //Splits the directory part by a space.
    StudentCode = directoryPart[0];
    FirstName = directoryPart[1];
    LastName = directoryPart[2];
}
```

Explanation: In the FromDirectory method, we extract StudentCode, FirstName and LastName filling the student object.

**d. Add a FromCSV method to extract data from a CSV into the Student model**

```
1 reference
public void FromCsv(string csvDataLine)
{
    string[] CsvDataLineParts = csvDataLine.Split(",", StringSplitOptions.None);
    DateOfBirthString = CsvDataLineParts[3];
    ImageData = CsvDataLineParts[4];
}
```

The FromCSV method has the DateOfBirthString and ImageData extracted from csv file.

**e. Add a ToCSV method that converts the contents of the Model to a CSV representation**

```
0 references
public string ToCSV()
{
    string result = $"{StudentCode},{FirstName},{LastName},{DateOfBirthString},{ImageData},{Record}";
    return result;
}
```

The ToCSV method displays the content of the Model in the CSV representation.



## 4.Retrieve the Student information programmatically from each directory

1. Add each student into a List object programmatically as a Student Model from 3.1.a

1. HINT: List<Student>

```
List<Student> students = new List<Student>(); //Creates a list named students.
```

2. Fill each Student with data retrieved from the info.csv and myimage.jpg from within the student directory

```
Student student = new Student(); //Creates a student object.
student.FromDirectory(directory); //Calls the FromDirectory method.
var fileBytes = FTP.DownloadFileBytes(Constants.FTP.BaseUrl + "/" + directory + "/info.csv"); //Downloads the bytes of each file in the directory.
string infoCsvData = Encoding.UTF8.GetString(fileBytes, 0, fileBytes.Length);
string[] lines = infoCsvData.Split("\r\n", StringSplitOptions.RemoveEmptyEntries); //Splits by return-carriage
student.FromCsv(lines[1]); //inputs the second column of the csv file which is the data record.
students.Add(student); //Adds student to students list.
```

The above code creates a student object and adds each student object to the students list.

## 2.Output the information to the Console using the modified ToString() function

```
public override string ToString()
{
    return $"{StudentCode}-{FirstName},{LastName},{DateOfBirthString},{ImageData},{Record}";
}
```

When the ToString method is called in the program.cs, we get the below output.

```
083100937-Leah,Stepanek,04/11/1965,False
200289345-Carolyn,Knight,05/20/1987,False
200335615-Chris,Dyck,11/29/1971,False
200443749-Binaya,Gurung,08/06/1995,False
200449414-Jiyoung,Sohn,08/04/1980,False
200449869-Sucheta,Karande,04/11/1996,False
200450776-Jenisha,Thummar,21-10-1998,False
200452357-Falak,Ghoda,17-04-1998,False
200454993-Ankur,Singh,01/08/1992,False
200454997-Sangeetha,Jayachandran,06/25/1998,False
200455930-Meenu,Rajan,,False
200458890-KelvinJose,Pinto,06/03/1997,False
200459092-Rajdeep,Kaur,03/03/1998,False
200460055-MehmetOzan,Kaya,29/06/1995,False
200463142-Piyush,Tyagi,11/14/1996,False
200464602-Dixit,Ohri,02/04/1995,False
200464960-Shivansh,Desai,03/10/1994,False
200467080-Manoj,Gottumukkala,18/10/97,True
200467506-Janina,Cortez,11/21/1994,False
200467632-Meet,Patel,11/12/1998,False
200468325-Abdullah,Mohammed,03/21/1977,False
200468841-AndrewKurian,Jacob,05/05/1997,False
200469298-Tomasz,Szumski,01/07/57,False
200470130-Osahon,Ighodaro, 11 Dec 1986,False
200470693-NnennaGrace,Alphonsus,12/12/1990,False
200471222-Sophie,Kellman,19810101,False
200471292-Danhong,Tang,11/30/1979,False
200471940-Karish,Thangarajah,3/6/1997,False
200473131-Sol,Jun,06/30/1997,False
200473709-David,Mascioli,09/21/1972,False
```

### 3. Output the information to the Console using the modified ToCSV() function.

```
public string ToCSV()
{
    string result = $"{StudentCode},{FirstName},{LastName},{DateOfBirthString},{Record}";
    return result;
}
```

```
string output = student.ToCSV();
Console.WriteLine(output);
```

The above two codes displays the information on the console.

```
083100937,Leah,Stepanek,04/11/1965,False
200289345,Carolyn,Knight,05/20/1987,False
200335615,Chris,Dyck,11/29/1971,False
200443749,Binaya,Gurung,08/06/1995,False
200449414,Jiyoung,Sohn,08/04/1980,False
200449869,Sucheta,Karande,04/11/1996,False
200450776,Jenisha,Thummar,21-10-1998,False
200452357,Falak,Ghoda,17-04-1998,False
200454993,Ankur,Singh,01/08/1992,False
200454997,Sangeetha,Jayachandran,06/25/1998,False
200455930,Meenu,Rajan,,False
200458890,KelvinJose,Pinto,06/03/1997,False
200459092,Rajdeep,Kaur,03/03/1998,False
200460055,MehmetOzan,Kaya,29/06/1995,False
200463142,Piyush,Tyagi,11/14/1996,False
200464602,Dixit,Ohri,02/04/1995,False
200464960,Shivansh,Desai,03/10/1994,False
200467080,Manoj,Gottumukkala,18/10/97,True
200467506,Janina,Cortez,11/21/1994,False
200467632,Meet,Patel,11/12/1998,False
200468325,Abdullah,Mohammed,03/21/1977,False
200468841,AndrewKurian,Jacob,05/05/1997,False
200469298,Tomasz,Szumski,01/07/57,False
200470130,Osahon,Ighodaro, 11 Dec 1986,False
200470693,NnennaGrace,Alphonsus,12/12/1990,False
200471222,Sophie,Kellman,19810101,False
200471292,Danhong,Tang,11/30/1979,False
200471940,Karish,Thangarajah,3/6/1997,False
200473131,Sol,Jun,06/30/1997,False
200473709,David,Mascioli,09/21/1972,False
```

## 5.

### 1. Display the results of the following Aggregate functions:

#### a. Count of all items in the List

```
int count = students.Count(); //Counts the number of students in the list.
Console.WriteLine($"The list contain {count} students"); //Displays the count.
```

The first statement counts the number of elements in the students list. The second statement displays the count.

Output:

```
The list contain 30 students
```

## b.StartsWith and display the count of items

```
foreach (var i in students)
{
    str1 = i.FirstName;

    if (str1.StartsWith('M'))
    {
        Console.WriteLine(i);
        countabc = countabc + 1;
    }
}

Console.WriteLine($"The number of students whose name starts with M is {countabc}");
```

Explanation: After declaring str1 and countabc, each element of the students list is iterated to count the number of instances whose first name startswith 'M'. The output of the above program is

```
200455930-Meenu,Rajan,,False
200460055-MehmetOzan,Kaya,29/06/1995,False
200467080-Manoj,Gottumukkala,18/10/97,True
200467632-Meet,Patel,11/12/1998,False
The number of students whose name starts with M is 4
```

In the console, the number of records that have firstname to start with 'M' and their count is displayed.

## c.Contains and display the count of items

### Code:

```
Student me = new Student
{
    StudentCode = "200467080",
    FirstName = "Manoj",
    //LastName = "Gottumukkala",
    //DateOfBirthString = "18/10/97"
};

bool listContains = students.Contains(me); //Checks if me object is in the students list.

if (listContains == true)
{
    Console.WriteLine($"Found {me}"); //If it contains, then displays the content.
}
else
{
    Console.WriteLine($"Could not find {me}");
}
```

## d. Write a query to find your student record using Find or SingleOrDefault in the list

- (a). Use this to mark "your record" in the list using MyRecord property of Student
- (b). Use List.Find to find your record and set the MyRecord property to true



```

Student abc = students.Find(x => x.FirstName == "Manoj"); //Checks if the first name is manoj from the students list.
//Checks if abc is not null and the value of counta (The reason for counta=1 is, the loop only triggers for the first occurrence of my name in the list)
if (abc != null && counta==1)
{
    student.Record = true; //Sets my record to true when my name enters the list for the first time.
    counta = counta + 1;
}
else
{
    student.Record = false; //Sets false to other directories.
}
string output = student.ToCSV();
Console.WriteLine(output);

```

Explanation: The reason for considering counta is as we are searching from the list(students), every student record is set to true after my name inserted into the list. So, I have used counta to trigger only my record.

Output:

```

083100937-Leah,Stepanek,04/11/1965,False
200289345-Carolyn,Knight,05/20/1987,False
200335615-Chris,Dyck,11/29/1971,False
200443749-Binaya,Gurung,08/06/1995,False
200449414-Jiyoung,Sohn,08/04/1980,False
200449869-Sucheta,Karande,04/11/1996,False
200450776-Jenisha,Thummar,21-10-1998,False
200452357-Falak,Ghoda,17-04-1998,False
200454993-Ankur,Singh,01/08/1992,False
200454997-Sangeetha,Jayachandran,06/25/1998,False
200455930-Meenu,Rajan,,False
200458890-KelvinJose,Pinto,06/03/1997,False
200459092-Rajdeep,Kaur,03/03/1998,False
200460055-MehmetOzan,Kaya,29/06/1995,False
200463142-Piyush,Tyagi,11/14/1996,False
200464602-Dixit,Ohri,02/04/1995,False
200464960-Shivansh,Desai,03/10/1994,False
200467080-Manoj,Gottumukkala,18/10/97,True
200467506-Janina,Cortez,11/21/1994,False
200467632-Meet,Patel,11/12/1998,False
200468325-Abdullah,Mohammed,03/21/1977,False
200468841-AndrewKurian,Jacob,05/05/1997,False
200469298-Tomasz,Szumski,01/07/57,False
200470130-Osahon,Ighodaro, 11 Dec 1986,False
200470693-NnennaGrace,Alphonsus,12/12/1990,False
200471222-Sophie,Kellman,19810101,False
200471292-Danhong,Tang,11/30/1979,False
200471940-Karish,Thangarajah,3/6/1997,False
200473131-Sol,Jun,06/30/1997,False
200473709-David,Mascioli,09/21/1972,False

```

We can observe that my record is set to TRUE.

### e.Average Age of the students:

For this purpose, I have created an average method in the student.cs and called that method in the program.cs. The sum has the sum of all the ages and Count2 has the count of number of students.

```
public static int average()
{
    avg = sum / Count2;
    Console.WriteLine(avg);
    return avg;
}

int count2 = Assign3.Model.Student.average(); //Finds the average age of the students.
Console.WriteLine($"The average age of students is {count2}");
```

With the help of two programs, we get the output as

```
The average age of students is 31
```

### f.Highest (Max) Age

Created highest function in the students.cs and calling it in the program.cs

```
public static int highest()
{
    high = abc;
    return (high);
}

int age = x - (1900 + y); //Adding 1900 to y and subtracting from 2020. i.e 2020-1997
//loop to find the maximum age.
if (abc < age)
{
    abc = age;
}

int count3 = Assign3.Model.Student.highest(); //Finds the maximum age of the students.
Console.WriteLine($"The maximum age of the students is {count3}");
```

After running the above programs, we get the output as

```
The maximum age of the students is 63
```

### g.Lowest (Min) Age

```
The minimum age of the students is 22
```

The minimum age of the students is 22.

**2. Output the contents of the List object to a CSV file. Indicate your student record using the result of the query in 5.1.d. If you have done 5.1.d correctly MyRecord should already be set to true.**

```
public string ToCSV()
{
    string docPath = Environment.GetFolderPath(Environment.SpecialFolder.Desktop);

    using (StreamWriter outputFile = new StreamWriter(Path.Combine(docPath, "students.csv"), true)) //true indicates it supports overwriting.
    {
        outputFile.WriteLine(StudentCode.Trim() + "," + FirstName.Trim() + "," + LastName.Trim() + "," + Record);
    }
    string result = $"{StudentCode},{FirstName},{LastName},{DateOfBirthString},{Record}";
    return result;
}
```

Explanation: docPath gets the path to the Desktop. StreamWriter is used to continuous append the data to students.csv file. The writeline adds the content to the file.

	A	B	C	D	E
1	83100937	Leah	Stepanek	FALSE	
2	2E+08	Carolyn	Knight	FALSE	
3	2E+08	Chris	Dyck	FALSE	
4	2E+08	Binaya	Gurung	FALSE	
5	2E+08	Jiyoung	Sohn	FALSE	
6	2E+08	Sucheta	Karande	FALSE	
7	2E+08	Jenisha	Thummar	FALSE	
8	2E+08	Falak	Ghoda	FALSE	
9	2E+08	Ankur	Singh	FALSE	
10	2E+08	Sangeetha	Jayachandi	FALSE	
11	2E+08	Meenu	Rajan	FALSE	
12	2E+08	KelvinJose	Pinto	FALSE	
13	2E+08	Rajdeep	Kaur	FALSE	
14	2E+08	MehmetOz	Kaya	FALSE	
15	2E+08	Piyush	Tyagi	FALSE	
16	2E+08	Dixit	Ohri	FALSE	
17	2E+08	Shivansh	Desai	FALSE	
18	2E+08	Manoj	Gottumukk	TRUE	
19	2E+08	Janina	Cortez	FALSE	
20	2E+08	Meet	Patel	FALSE	
21	2E+08	Abdullah	Mohamme	FALSE	
22	2E+08	AndrewKur	Jacob	FALSE	
23	2E+08	Tomasz	Szumski	FALSE	
24	2E+08	Osahon	Ighodaro	FALSE	
25	2E+08	NnennaGr	Alphonsus	FALSE	
26	2E+08	Sophie	Kellman	FALSE	
27	2E+08	Danhong	Tang	FALSE	
28	2E+08	Karish	Thangaraja	FALSE	

This is the format of the csv file created with my name set to TRUE.

3. Output the contents of the List object to a JSON file. Indicate your student record using the result of the query in 5.1.d. If you have done 5.1.d correctly MyRecord should already be set to true.

**Code:**

```
foreach (var i in students)
{
    JObject stud = new JObject(
        new JProperty("StudenCode", i.StudenCode),
        new JProperty("FirstName", i.FirstName),
        new JProperty("LastName", i.LastName),
        new JProperty("Record", i.Record));

    // serialize JSON directly to a file
    using (StreamWriter file = File.AppendText("C:\\Users\\Owner\\Desktop\\students.json"))
    {
        JsonSerializer serializer = new JsonSerializer();
        serializer.Serialize(file, stud);
    }
}
```

**Sample of the output:**

```
{
  "StudenCode": "083100937",
  "FirstName": "Leah",
  "LastName": "Stepanek",
  "Record": false
}
{
  "StudenCode": "200289345",
  "FirstName": "Carolyn",
  "LastName": "Knight",
  "Record": false
}
{
  "StudenCode": "200335615",
  "FirstName": "Chris",
  "LastName": "Dyck",
  "Record": false
}
{
  "StudenCode": "200443749",
  "FirstName": "Binaya",
  "LastName": "Gurung",
  "Record": false
}
{
  "StudenCode": "200449414",
  "FirstName": "Jiyoung",
  "LastName": "Sohn",
  "Record": false
}
```

4. Output the contents of the List object to an XML file. Indicate your student record using the result of the query in 5.1.d. If you have done 5.1.d correctly MyRecord should already be set to true.

```
string docPath = Environment.GetFolderPath(Environment.SpecialFolder.Desktop);
XmlTextWriter textWriter = new XmlTextWriter("C:\\Users\\Owner\\Desktop\\students.xml", null); //Takes the file path.
textWriter.WriteStartDocument(); //starts writing to the document.
textWriter.WriteStartElement("StudentsInfo"); //Starts the element.

foreach (var i in students)
{
    textWriter.WriteStartElement("Student");
    textWriter.WriteAttributeString("Studentcode", i.StudentCode);
    textWriter.WriteString(i.FirstName);
    textWriter.WriteString(" ");
    textWriter.WriteString(i.LastName);
    textWriter.WriteString(" ");
    textWriter.WriteString(i.Record.ToString());
    textWriter.WriteEndElement();
}
textWriter.WriteEndDocument();
textWriter.Close();
```

This is the code I used to implement the task. The students.xml file when opened in visual studio looks like this with no errors.

```
<?xml version="1.0"?>
<StudentsInfo>
  <Student Studentcode="083100937">Leah Stepanek False</Student>
  <Student Studentcode="200289345">Carolyn Knight False</Student>
  <Student Studentcode="200335615">Chris Dyck False</Student>
  <Student Studentcode="200443749">Binaya Gurung False</Student>
  <Student Studentcode="200449414">Jiyoung Sohn False</Student>
  <Student Studentcode="200449869">Sucheta Karande False</Student>
  <Student Studentcode="200450776">Jenisha Thummar False</Student>
  <Student Studentcode="200452357">Falak Ghoda False</Student>
  <Student Studentcode="200454993">Ankur Singh False</Student>
  <Student Studentcode="200454997">Sangeetha Jayachandran False</Student>
  <Student Studentcode="200455930">Meenu Rajan False</Student>
  <Student Studentcode="200458890">KelvinJose Pinto False</Student>
  <Student Studentcode="200459092">Rajdeep Kaur False</Student>
  <Student Studentcode="200460055">MehmetOzan Kaya False</Student>
  <Student Studentcode="200463142">Piyush Tyagi False</Student>
  <Student Studentcode="200464602">Dixit Ohri False</Student>
  <Student Studentcode="200464960">Shivansh Desai False</Student>
  <Student Studentcode="200467080">Manoj Gottumukkala True</Student>
  <Student Studentcode="200467506">Janina Cortez False</Student>
  <Student Studentcode="200467632">Meet Patel False</Student>
  <Student Studentcode="200468325">Abdullah Mohammed False</Student>
  <Student Studentcode="200468841">AndrewKurian Jacob False</Student>
  <Student Studentcode="200469298">Tomasz Szumski False</Student>
  <Student Studentcode="200470130">Osahon Ighodaro False</Student>
  <Student Studentcode="200470693">NnennaGrace Alphonsus False</Student>
  <Student Studentcode="200471222">Sophie Kellman False</Student>
  <Student Studentcode="200471292">Danhong Tang False</Student>
  <Student Studentcode="200471940">Karish Thangarajah False</Student>
  <Student Studentcode="200473131">Sol Jun False</Student>
  <Student Studentcode="200473709">David Mascioli False</Student>
```

```
File Edit Format View Help
<?xml version="1.0"?><StudentsInfo><Student Studentcode="083100937">Leah Stepanek False</Student><Student Studentcode="200289345">Carolyn Knight False</Student><Student Studentcode="2003356">
tudent><Student Studentcode="200464960">Shivansh Desai False</Student><Student Studentcode="200467080">Manoj Gottumukkala True</Student><Student Studentcode="200467506">Janina Cortez False</Student>
```

We can observe that my record is set to true.

## 5.Move the CSV, JSON and XML files programmatically into your FTP directory. Name them students.csv, students.json, and students.xml

### Uploading csv file code:

```
string localUploadFilePath = @"C:\Users\Owner\Desktop\students.csv";
string remoteUploadFileDestination = "/200467080 Manoj Gottumukkala/students.csv";

Console.WriteLine(FTP.UploadFile(localUploadFilePath, Constants.FTP.BaseUrl + remoteUploadFileDestination));
```

### Uploading xml file code:

```
string localUploadFilePath = @"C:\Users\Owner\Desktop\students.xml";
string remoteUploadFileDestination = "/200467080 Manoj Gottumukkala/students.xml";

Console.WriteLine(FTP.UploadFile(localUploadFilePath, Constants.FTP.BaseUrl + remoteUploadFileDestination));
```

### Uploading json file code:

```
string localUploadFilePath = @"C:\Users\Owner\Desktop\students.json";
string remoteUploadFileDestination = "/200467080 Manoj Gottumukkala/students.json";

Console.WriteLine(FTP.UploadFile(localUploadFilePath, Constants.FTP.BaseUrl + remoteUploadFileDestination));
```