ATILIM UNIVERSITY
FACULTY OF ENGINEERING

DEPARTMENT OF SOFTWARE ENGINEERING

SE 399 SUMMER PRACTICE I REPORT

Prepared By
Samet Çolak
20244710030

Oct. 14, 2023

Table of contents

1. LIST OF TABLES AND FIGURES

2. INTRODUCTION

Summer internship offers students the opportunity to experience the work they will do before graduating from their department. Engineering students must complete 2 mandatory summer internships before graduating. I am studying software engineering and I did my first internship this summer between 31/07/2023 and 25/08/2023. This was my first work experience in the software industry. This internship, which lasted 20 working days, gave me different experiences in many subjects. I had the opportunity to see the working environments of software engineers and to see the advantages and difficulties of software engineering. This internship report consists first of the place of the company Duru Bilişim, where I worked, in the sector, what I did during my internship at this company, my daily routine, the advantages and disadvantages of the department I have experienced, and finally the conclusion and suggestions.

3. DESCRIPTION OF THE COMPANY

Figure-1: Company Logo

Duru Information Technologies Ltd. Ltd. With its vision that direct and pioneer the sector, it has been developing software solutions that will facilitate the functioning of Building Inspection Companies, Building Inspection Laboratories, and Municipalities since 2000. Duru Information Technologies Ltd. Ltd. He stepped into the sector in 2000 by making software for the building inspection sector.

Its goal is to eliminate mistakes and disruptions in the construction industry by using technology, to contribute to its progress in the right direction, and to prepare the necessary software in this direction.

The studies carried out in line with this goal are as follows:

2000 - Turkey's first Building Inspection application - (YapDen 2000)

2002 - Turkey's first Yapı Laboratory application (YapDen Laboratory),

2006 - Municipal Zoning License and Building Inspection Application (Yapden Municipality), integrated with TURKSTAT and YDS,

2011 - Turkey's first Building Inspection Work Distribution Applications (Denizli, Çorum, Konya),

2014 - Turkey's first Maks-integrated Municipal Zoning Application (D-MAKS),

2015 - Turkey's first QR code Concrete Report Application

2018 - Turkey's first Electronic Concrete Monitoring System compatible Construction Laboratory Application (YapDen EBİS)

2019 - Ministry of Environment and Urbanization National Building Inspection System Work Distribution Application Consultancy

2019 - Turkey's first UYDS compatible Building Inspection Application (YapDen UYDS)

2020 - Electronic Application and Tracking Application

2020 - Turkey's first Zoning Management System Application (D-İMAR)

The company has been operating in Hacettepe University Technocity since 2017.

Duru Information Technologies Ltd. Ltd. Since its establishment, it has been expanding its technical infrastructure and serving its customers with the same excitement and same principles as the first day.

4. DESCRIPTION OF THE WORK DONE

4.1. Habituation Process

The first week of my internship was an adjustment period for me. During this process, I had the opportunity to meet the employees of the company. I met Mr. Yusuf, who works as a software test specialist in our company. Mr. Yusuf told me about the importance and place of testing in software. Testing is very important in determining software quality. Testing in software is divided into two: Automation and manual testing. I had the opportunity to see a few tests performed by Mr. Yusuf. I learned C# in the first week because the applications made in the company were made with C#. That's why my Internship coach, Mr. Ercüment, wanted me to learn the C# language and helped me in this process. After learning C#, he asked me to make two applications with the information I learned. The first of these

applications is the Folder cleaning application and the other is the Book storge application. These applications gave me a lot of experience and the opportunity to repeat what I learned in C#.

4.2. Folder Cleaning App

4.2.1. Description of the application

This application deletes the remaining files in a certain format in all folders under a certain folder, leaving the desired number of files. In this way, it always keeps current data and deletes old data. While doing this application, I learned and used the recursive function logic. While making this application, I thought that it was necessary to use a recursive function and implemented it. Finally, it creates a log file and writes the names of the deleted files to this file. I leave some photos about the project below.

```
1 reference
private static void NullArgs()
{
    string rootFolderPath;
    Console.WriteLine("Ana dosyanın konumunu giriniz :");
    rootFolderPath= $"{Console.ReadLine()}";
    Console.WriteLine("Silmek istediğniz dosya uzantısını (txt) giriniz:");
    string fileExtention = Console.ReadLine();
    Console.WriteLine("Kalmasını istediğniz dosya sayısı:");
    int filesKeep = Convert.ToInt16(Console.ReadLine());
    Console.WriteLine("Log için dosya yolu girmek istiyor musunuz(otamatik .exe dosyasının yanına kurulur)(evet/hayır)");
    string logFile = "";
    string cevap= Console.ReadLine();
    if (cevap=="evet")
    {
        Console.WriteLine("Log için dosya uzantısı giriniz;");
        logFile= Console.ReadLine();
    }
    else if (cevap=="hayır")
    {
        string dizin = Path.GetDirectoryName(Assembly.GetExecutingAssembly().Location);
        DateTime now = DateTime.Now;
        string dosyaAdı = $"Log {now.ToShortDateString()}.txt";
        string dosyaYolu = Path.Combine(dizin, dosyaAdı);
        using (FileStream dosya = File.Create(dosyaYolu))
        {
            Console.WriteLine("Dosya oluşturuldu: " + dosyaYolu);
        }
         logFile =dosyaYolu;
    }
    else
    {
        Console.WriteLine("Yanlış giriş yapıldı.");
    }
    int counter = 0;
    ProcessFolders(rootFolderPath, filesKeep, fileExtention, ref counter, ref logFile);
    Console.WriteLine("silinen dosya sayısı: " + counter);
    Console.ReadKey();

}
```

Figure-2: Folder Cleaning App Code-1

```
3 references
static void ProcessFolders(string folderPath, int maxFilesToKeep, string fileOfExtention, ref int _counter,ref string logDosyaAdı)
{
    try
    {
        if (!fileOfExtention.StartsWith("."))
            fileOfExtention = "." + fileOfExtention;

        var directoryInfo = new DirectoryInfo(folderPath);
        var files = directoryInfo.GetFileSystemInfos("*" + fileOfExtention).OrderByDescending(t => t.LastWriteTime)
            .Skip(maxFilesToKeep)
            .Where(t => t.Extension == fileOfExtention)
            .Select(t => t.FullName).ToArray();
        LogWriterSubdirectory(folderPath, ref logDosyaAdı);
        foreach (var file in files)
        {
            LogWriterFile(file, ref logDosyaAdı);
            File.Delete(file);

            _counter++;
        }
        LogWriterSubdirectory(folderPath, ref logDosyaAdı);

        string[] subdirectories = Directory.GetDirectories(folderPath);
        foreach (string subdirectory in subdirectories)
        {
            ProcessFolders(subdirectory, maxFilesToKeep, fileOfExtention, ref _counter,ref logDosyaAdı);

        }
    }
    catch (Exception ex)
    {
        Console.WriteLine("Hata oluştu: " + ex.Message);
    }
}
```

Figure-3: Folder Cleaning App Code-2

4.2.2. Operation with Command Line

I configured a few codes to run this console application I made with Commad line. We can run this command line or cmd with some timer applications and clean the desired file regularly.

```
namespace çalıştır
{
    0 references
    internal class Program
    {
        0 references
        static void Main(string[] argv)
        {
            string rootFolderPath="";
            int filesKeep=0;
            string fileKeepString;
            string fileExtention="";
            string logFile="";
            if (argv != null && argv.Length > 0)
            {
                if (argv.Length == 4)
                {
                    int i;
                    for (i=0; i<4; i++)
                    {

                        string a = $"{argv[i]}";

                       string find=a.Substring(0,2);
                        switch (find)
                        {
                            case "-y":
                             rootFolderPath= a.Replace("-y=", "");
                               break;
                            case "-a":
                                fileKeepString=a.Replace("-a=", "");
                                filesKeep = Convert.ToInt32(fileKeepString);
                                break;
                            case "-u":
                                fileExtention = a.Replace("-u=", "");
                                break;
                            case "-l":
                                logFile = a.Replace("-l=", "");
                                break;
                            default:
                                Console.WriteLine("kod satırını kontrol edin yanlış yazılmış!!!");
                                break;
                        }
```

Figure-4: Command Line-1

```
            }


        }
        else
        {
            Console.WriteLine("hatalı argüman girişi ! Doğru şekilde 4 argüman giriniz()");
        }
          if(logFile.Length == 0)
          {
            string dizin = Path.GetDirectoryName(Assembly.GetExecutingAssembly().Location);
            DateTime now = DateTime.Now;
            string dosyaAdı = $"Log {now.ToShortDateString()}.txt";
            string dosyaYolu = Path.Combine(dizin, dosyaAdı);
            using (FileStream dosya = File.Create(dosyaYolu))
            {
                Console.WriteLine("Dosya oluşturuldu: " + dosyaYolu);
            }
            logFile=dosyaYolu;
          }
        int counter = 0;
        ProcessFolders(rootFolderPath, filesKeep, fileExtention, ref counter, ref logFile);
        Console.WriteLine("silinen dosya sayısı: " + counter);
        Console.ReadKey();
    }
    else
    {
        NullArgs();
    }
```

Figure-5: Command Line-2

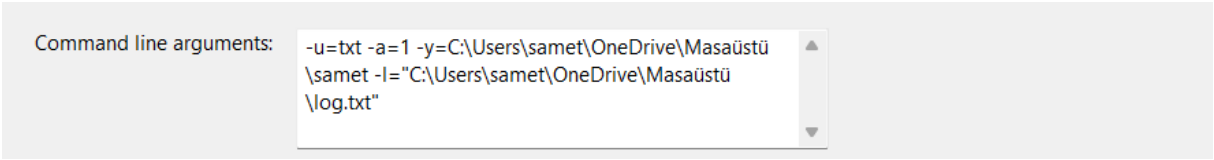


Figure-6: Command Line-3

4.3.Book Store App

4.3.1.Description of the Application

I made a Book store application with C#. This was a web Form application. On the other hand, I created an in-memory data and wrote a few books in it. I communicated with the Web API I created on localhost.

4.3.2.Logic of Web API

I developed Rest API with C#. I learned about http status codes.

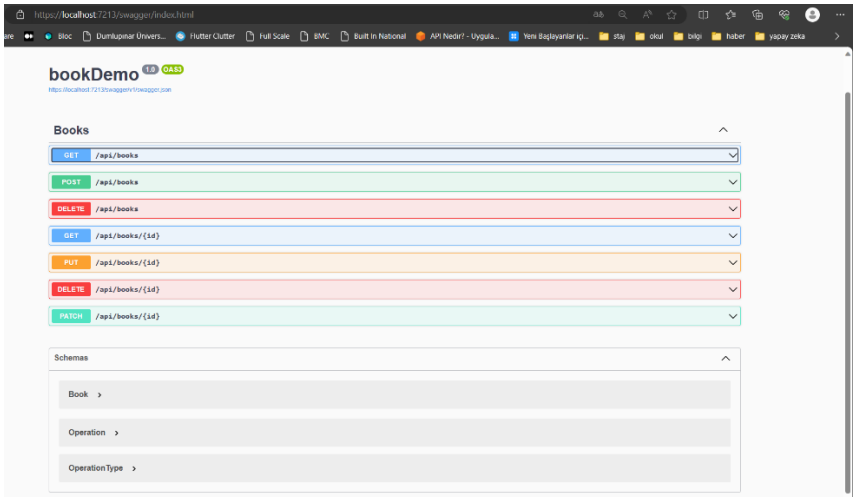


Figure-7: Swagger Web API

A. Get Request

```
[HttpGet]
0 references
public IActionResult GetAllBooks()
{
    var books = ApplicationContext.Books;
    return Ok(books);
}
[HttpGet("{id:int}")]
0 references
public IActionResult GetOneBooks([FromRoute (Name ="id")]int id)//bounding strongly
{
    // LINQ yapısı
    var book = ApplicationContext
        .Books
        .Where(b => b.Id.Equals(id))
        .SingleOrDefault();
    if(book == null)
       return NotFound();//404
    return Ok(book);//200
}
```

B. Post Request

```
[HttpPost]
0 references
public IActionResult CreatOneBook([FromBody] Book book)
{
    try
    {
        if(book == null)
            return BadRequest(); //400  istenilen tarzda olmaması

       ApplicationContext.Books.Add(book);
        return StatusCode(201, book);

    }
    catch (Exception ex)
    {
        return BadRequest(ex.Message);
    }
}
```

C. Put Request

```
[HttpPut("{id=int}")]
0 references
public IActionResult UpdateOneBook([FromRoute (Name ="id")] int id,
    [FromBody] Book book)
{
    //kitap varmı yok mu
    var entity =ApplicationContext
        .Books
        .Find(b=>b.Id.Equals(id));

    if(entity == null)
        return NotFound();//404

    //id konturolü
    if (id != book.Id)
        return BadRequest();//400
    ApplicationContext.Books.Remove(entity);
    book.Id = entity.Id;
    ApplicationContext.Books.Add(book);
    return Ok(book);
}
```

D. Delete Request

```
[HttpDelete]
0 references
public IActionResult DeleteAllBooks()
{
    ApplicationContext.Books.Clear();
    return NoContent(); //204
}

[HttpDelete("{id=int}")]
0 references
public IActionResult DeleteOneBooks([FromRoute(Name = "id")] int id)
{
    var entity = ApplicationContext
        .Books
        .Find(b => b.Id.Equals(id));

    if (entity == null)
        return NotFound(new
        {
            StatusCode =404,
            message=$"Book with id:{id} could not found"
        });//404

    ApplicationContext.Books.Remove(entity);
   return NoContent();
}
```

4.3.3. Creating the Web Form

I developed a Web Form with C#. I created the Web Form in a simple way using Visual Studio and Tools.

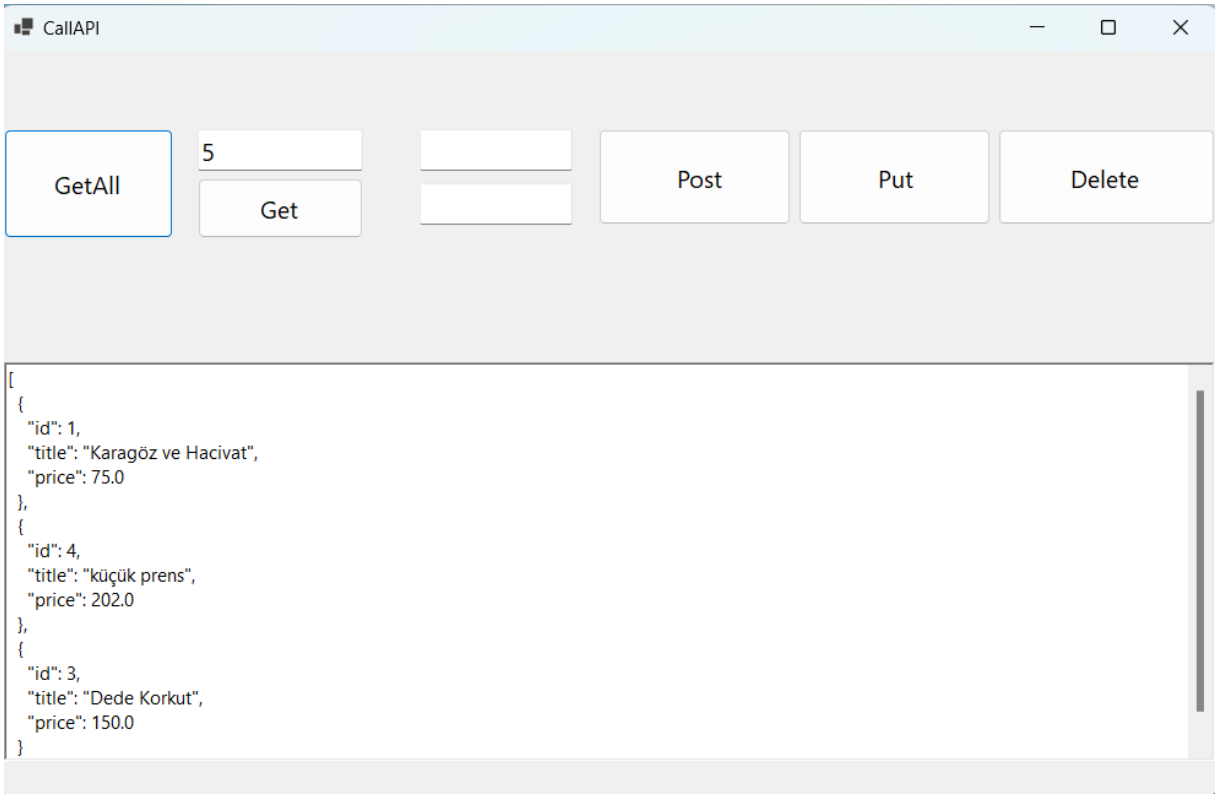
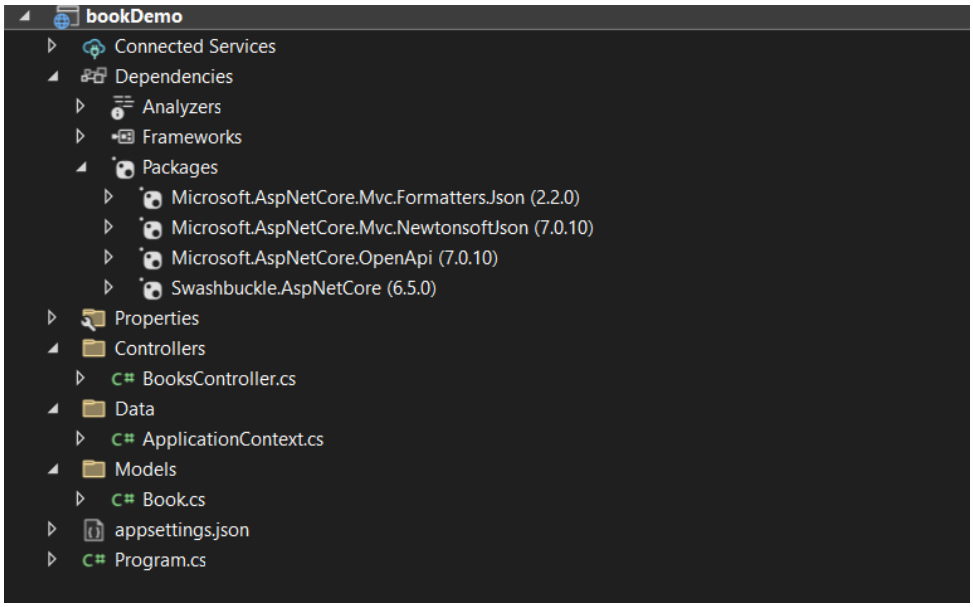


Figure-8: Web Form

4.3.4.MVC Structure



5.CONCLUSIONS AND SUGGESTIONS

Summer internship was very good for me. I met new people from the industry. They helped me chart a path for my future. In this summer internship, I learned how to develop web forms and rest APIs with C#. I learned about MVC and used this model in my project. I learned that although we do not spend any physical effort in software engineering, we spend a lot of mental energy. We do not have to know and experience every problem we encounter in advance. The employees there do not know everything, what is important is that they know the logic, read and analyze the documentation properly. In this way, we can solve every problem. The disadvantage of software engineering is sitting on the couch and staring at the computer all day long. I felt this thoroughly during the internship, but it is necessary to get used to it. As time goes by, people get used to sitting and staring at the screen for long periods of time. I'm looking forward to my 30-day internship next year.

6.REFERENCES

- https://www.durubilgisayar.com/