Software Design and Development

Module : Software Engineering |B9IS114

Module Leader : Harnik Dhoot

Assessment Type : Group

Student Name :Tapesh Patel |10375827

Ahmad Awayhan|1769801

Merlin Mary Joy |10369315

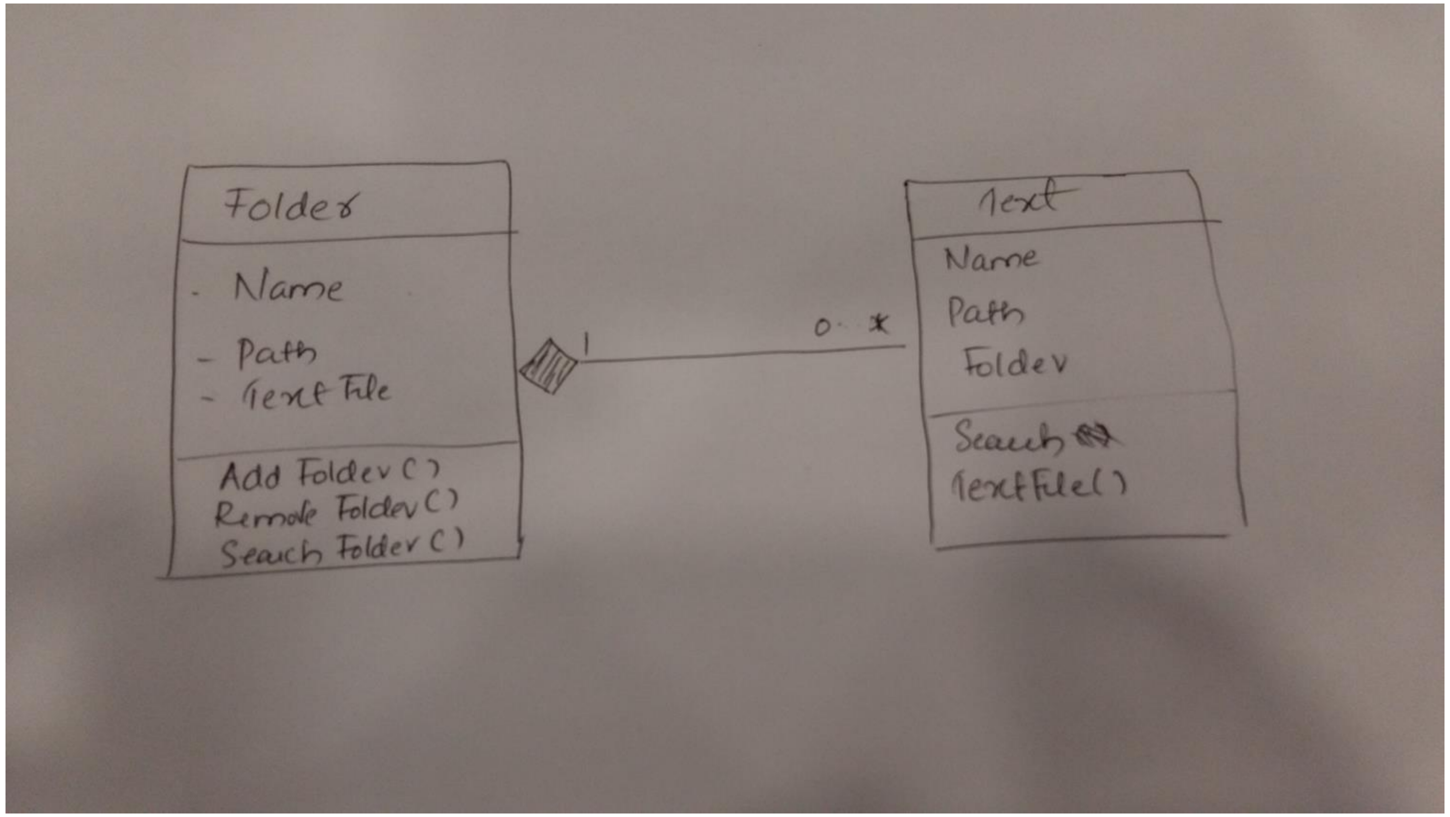
Bifurcation of work on individual basis

- Tapesh Patel (Question no 1, 2,11,12, 13, PPT, report, and UML)
- Ahmad Awayhan (Question no 3, 4 ,5, 9, 10,11)
- Merlin Mery Joy (Question no 7, 8, 12, 11)

Submission of work schedule

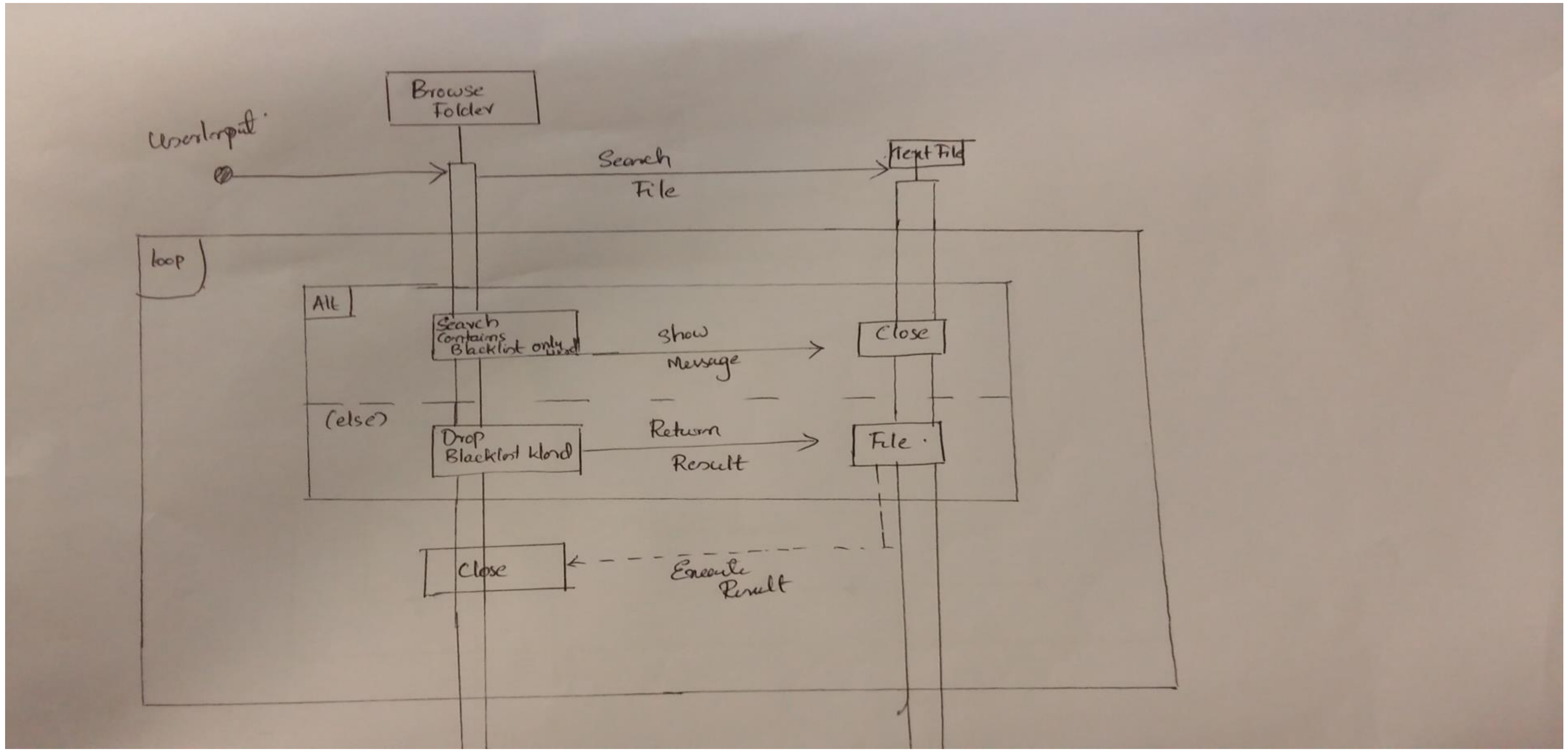
- Start Time : October
    - 10, 2017
    - 17, 2017
    - 24, 2017
    - 31, 2017
- Start Time : November
    - 7, 2017
    - 14,2017
    - 21,2017
    - 28,2017
- End Time : December 10, 2017
- Time and Place : Every Tuesday 3hours from 11 am to 2pm room number 3 and 4 library

UML Class Diagram



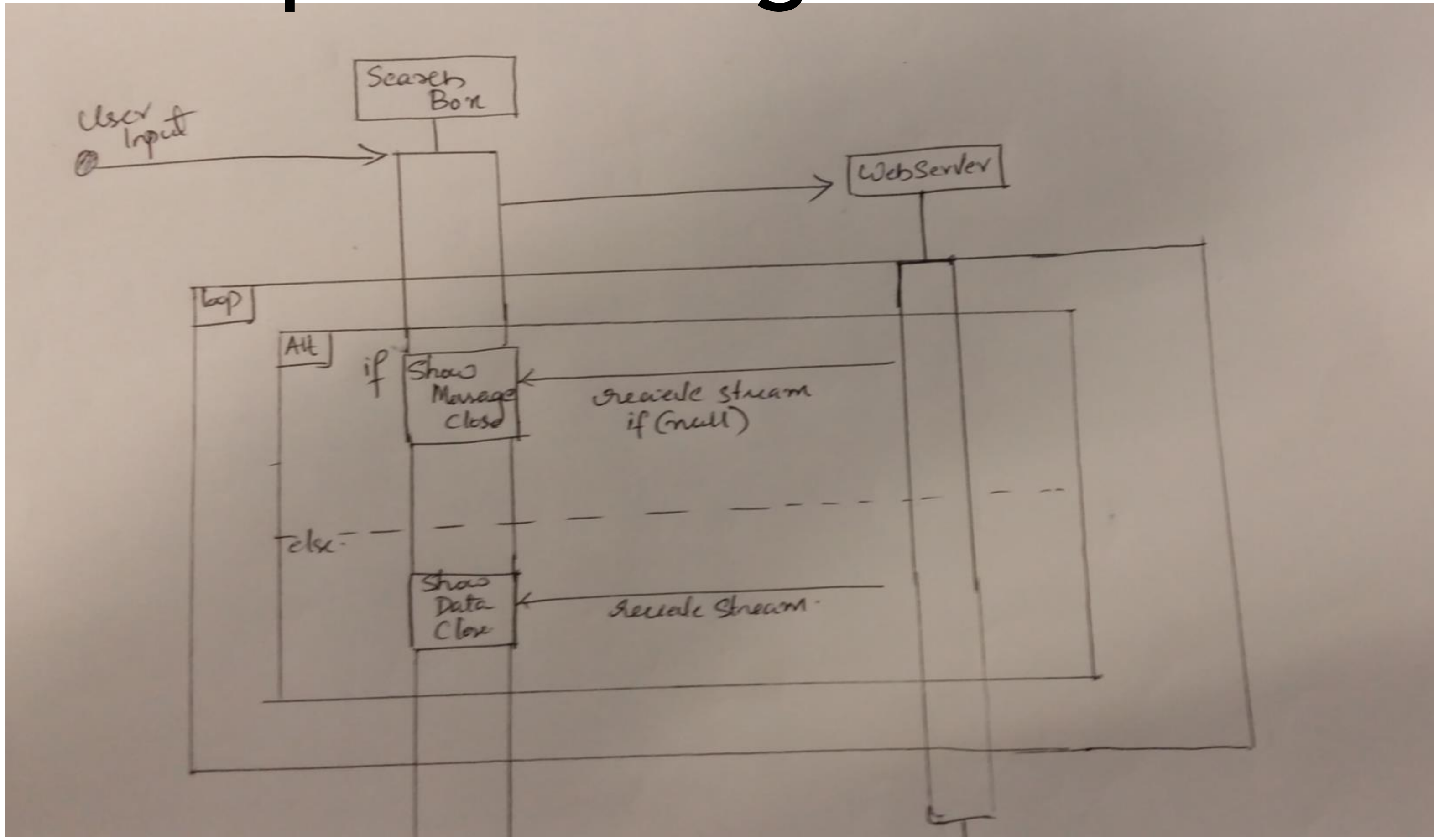
- Class diagram

UML Sequence Diagram



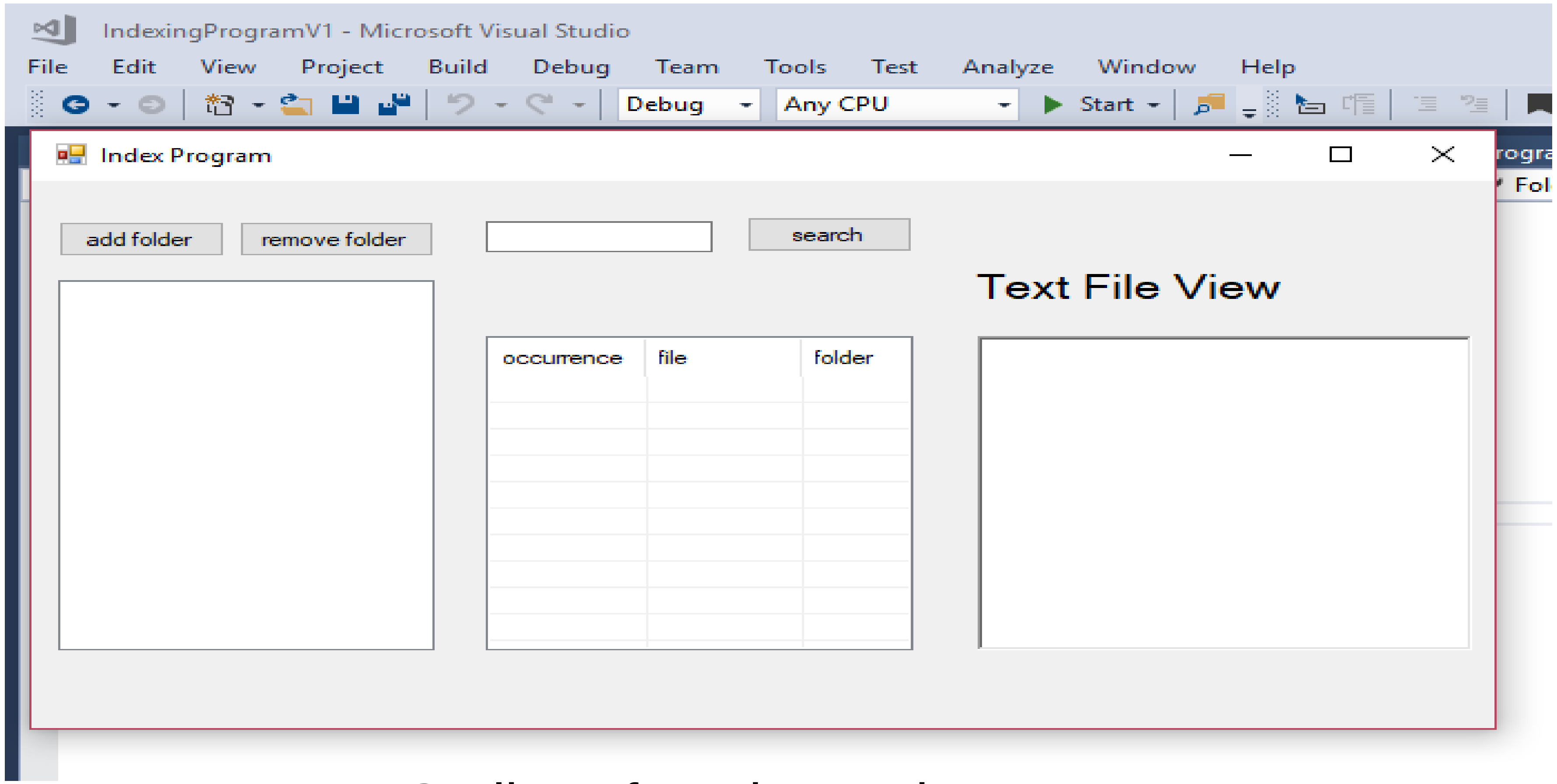
- **Indexing sequence diagram**

UML Sequence Diagram



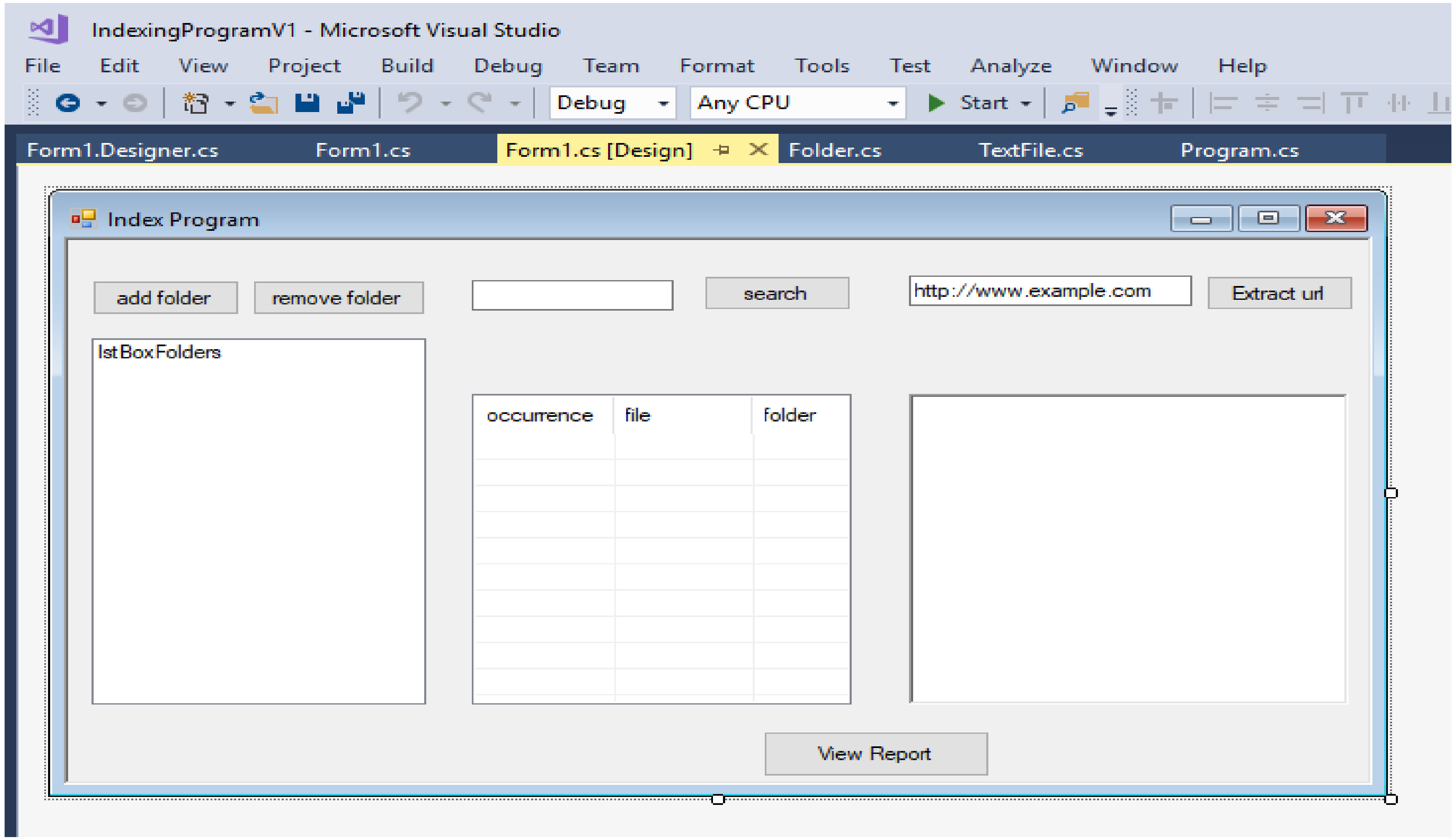
- HTML Sequence diagram

Implementation of text file code



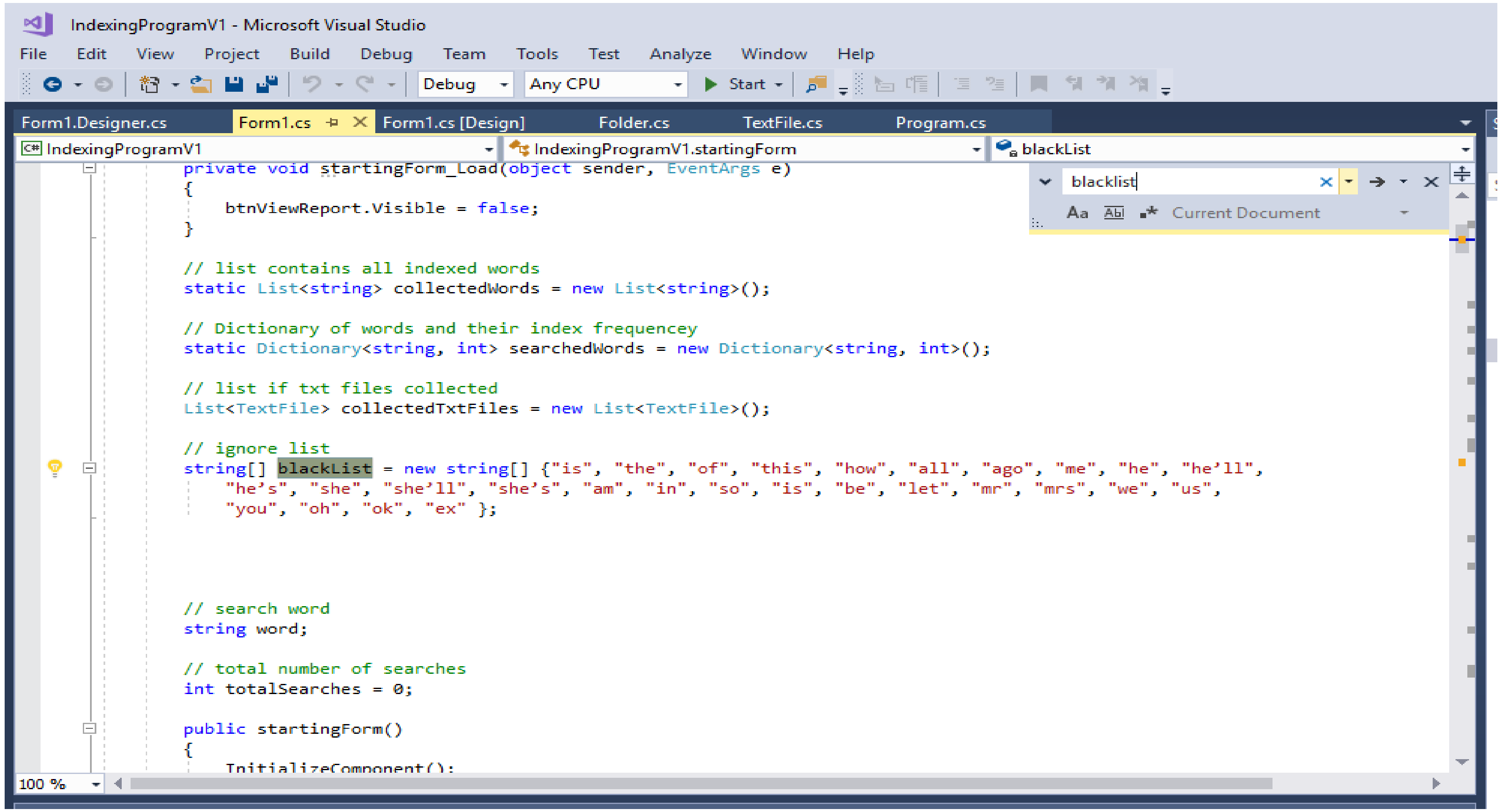
- Outline of Design and structure

Implementation of HTML code



- **Outline of design and structure**

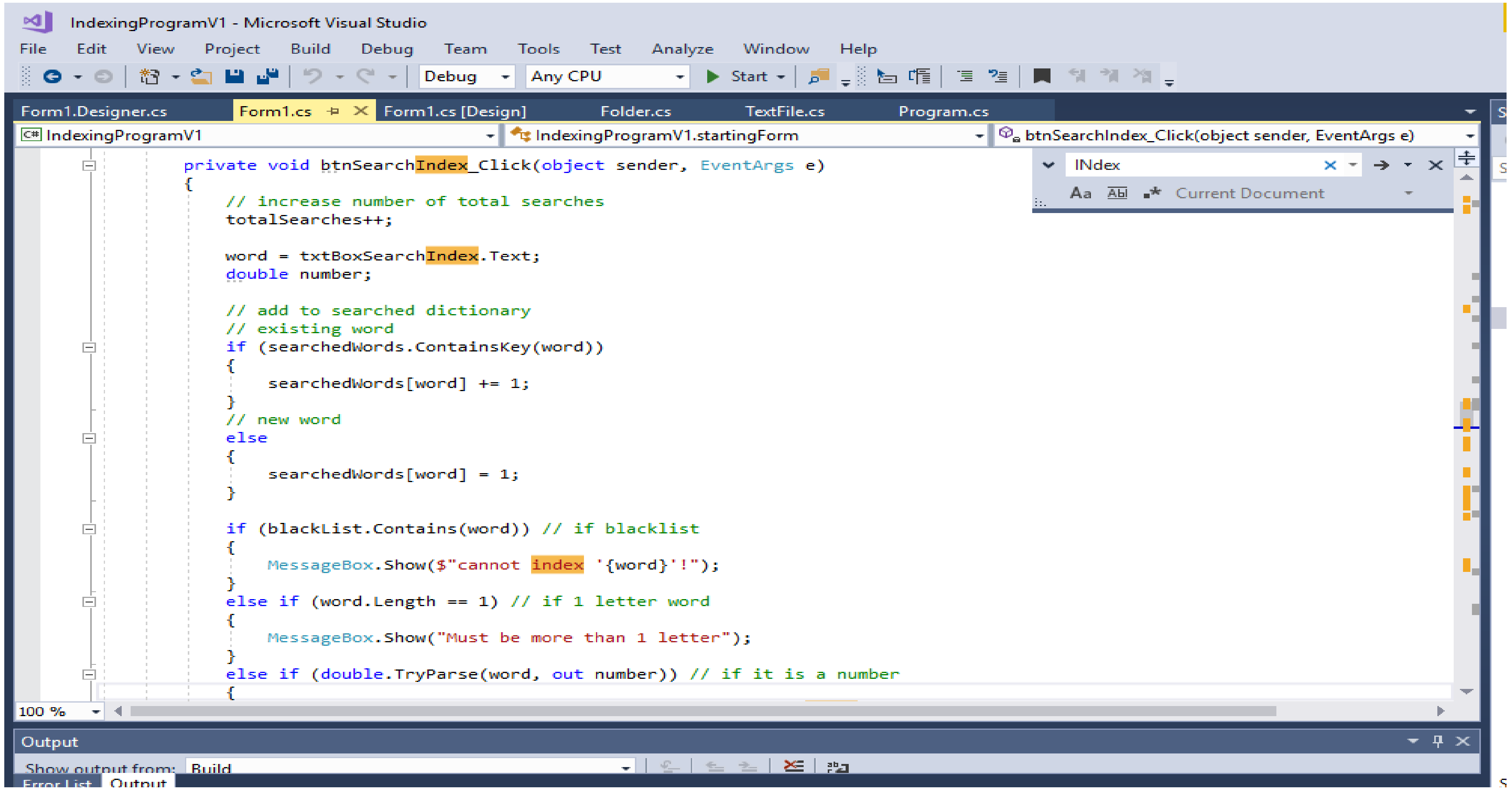
Index code (Blacklist)



IndexingProgramV1 - Microsoft Visual Studio
File Edit View Project Build Debug Team Tools Test Analyze Window Help
Debug Any CPU Start
Form1.Designer.cs Form1.cs Form1.cs [Design] Folder.cs TextFile.cs Program.cs
IndexingProgramV1 IndexingProgramV1.startingForm blackList
blacklist
Current Document
100 %

```
        private void startingForm_Load(object sender, EventArgs e)
        {
            btnViewReport.Visible = false;
        }

        // list contains all indexed words
        static List<string> collectedWords = new List<string>();

        // Dictionary of words and their index frequencey
        static Dictionary<string, int> searchedWords = new Dictionary<string, int>();

        // list if txt files collected
        List<TextFile> collectedTxtFiles = new List<TextFile>();

        // ignore list
        string[] blackList = new string[] {"is", "the", "of", "this", "how", "all", "ago", "me", "he", "he'll",
            "he's", "she", "she'll", "she's", "am", "in", "so", "is", "be", "let", "mr", "mrs", "we", "us",
            "you", "oh", "ok", "ex" };



        // search word
        string word;

        // total number of searches
        int totalSearches = 0;

        public startingForm()
        {
            InitializeComponent();
```

Indexing code (if typed 1 word)

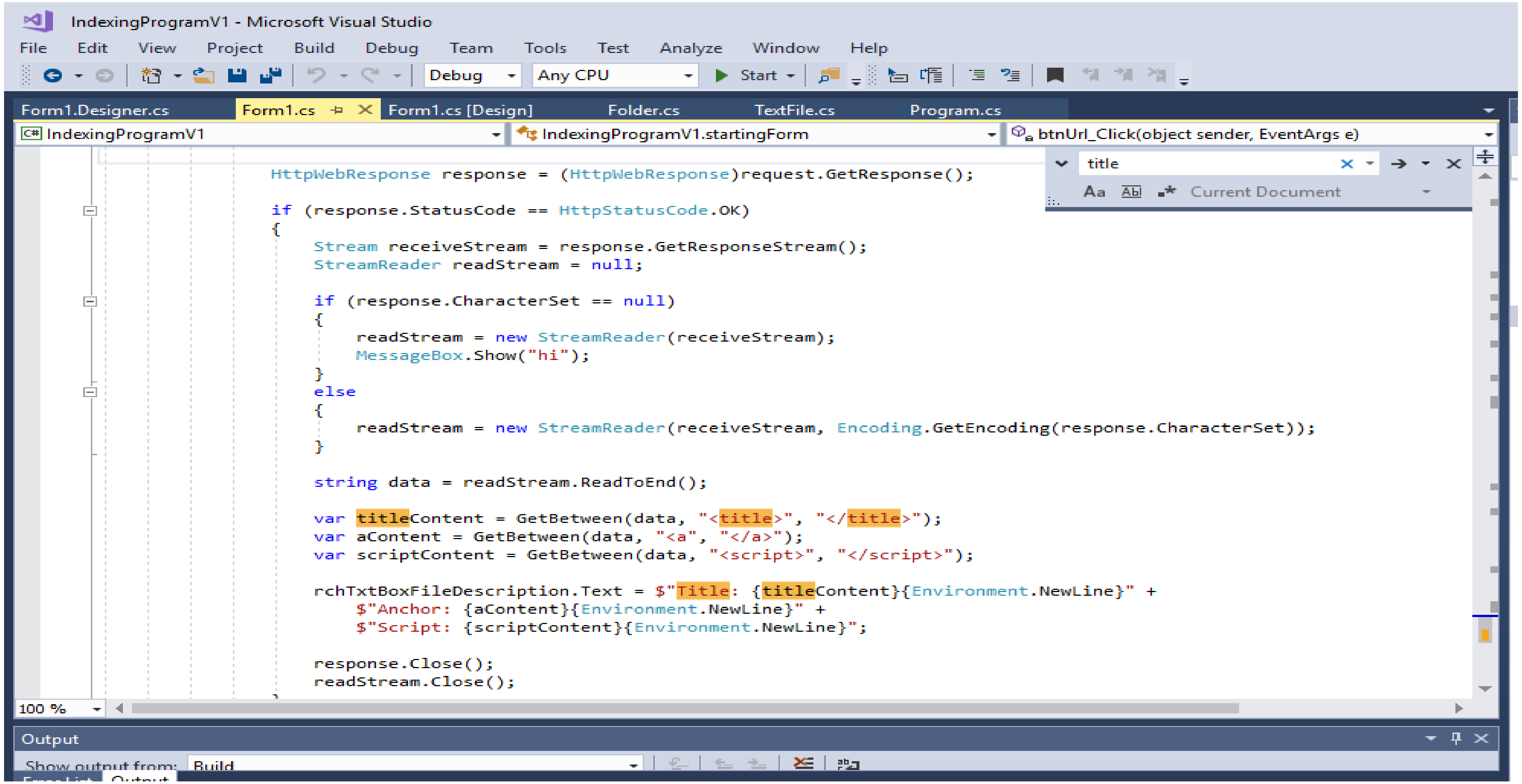


IndexingProgramV1 - Microsoft Visual Studio

Form1.cs

```
private void btnSearchIndex_Click(object sender, EventArgs e)
{
    // increase number of total searches
    totalSearches++;

    word = txtBoxSearchIndex.Text;
    double number;

    // add to searched dictionary
    // existing word
    if (searchedWords.ContainsKey(word))
    {
        searchedWords[word] += 1;
    }
    // new word
    else
    {
        searchedWords[word] = 1;
    }

    if (blackList.Contains(word)) // if blacklist
    {
        MessageBox.Show($"cannot index '{word}'!");
    }
    else if (word.Length == 1) // if 1 letter word
    {
        MessageBox.Show("Must be more than 1 letter");
    }
    else if (double.TryParse(word, out number)) // if it is a number
    {
```

HMTL Code



IndexingProgramV1 - Microsoft Visual Studio
File Edit View Project Build Debug Team Tools Test Analyze Window Help
Debug
Any CPU
Start
Form1.Designer.cs
Form1.cs
Form1.cs [Design]
Folder.cs
TextFile.cs
Program.cs
IndexingProgramV1
IndexingProgramV1.startingForm
btnUrl_Click(object sender, EventArgs e)
title
Current Document
100 %
Output
Show output from: Build

```
            HttpWebResponse response = (HttpWebResponse)request.GetResponse();

            if (response.StatusCode == HttpStatusCode.OK)
            {
                Stream receiveStream = response.GetResponseStream();
                StreamReader readStream = null;

                if (response.CharacterSet == null)
                {
                    readStream = new StreamReader(receiveStream);
                    MessageBox.Show("hi");
                }
                else
                {
                    readStream = new StreamReader(receiveStream, Encoding.GetEncoding(response.CharacterSet));
                }

                string data = readStream.ReadToEnd();

                var titleContent = GetBetween(data, "<title>", "</title>");
                var aContent = GetBetween(data, "<a", "</a>");
                var scriptContent = GetBetween(data, "<script>", "</script>");

                rchTxtBoxFileDescription.Text = $"Title: {titleContent}{Environment.NewLine}" +
                    $"Anchor: {aContent}{Environment.NewLine}" +
                    $"Script: {scriptContent}{Environment.NewLine}";

                response.Close();
                readStream.Close();
```

Limitations

- As per requirement of the assignment our group was unable to meet the need of assignment i.e. the implementation of binary file mentioned in Question 6.
- Also we are unable to implement the File system manager, but we tried to do until certain extent and the code for this is mentioned in the file with comment.