Project 1 - Web Crawler

Objective

In this project you will create a web crawler program and then deploy it to multiple App Service instances in Azure. The web crawler will open a file with a list of seeder sites. It will pick one at random and begin the process of scraping links from the HTML and then continuing to crawl on those new links. In addition to crawling links, the web crawler will keep a record of each ‘edge’ across its search. For each site visited, for example www.facebook.com, you will store all of the hosts that facebook linked to and every hour your web crawler will halt, generate a JSON of all the edges and the date visited, and write this JSON to Data Lake Storage in Azure. It will then restart from scratch (clear all of its memory, probably best just to destroy the object and create a new one), and repeat this process continually, generating data every hour from now to the end of Project 4.

Your web crawler is your telemetry layer, and this edges JSON data will serve as your source of data for the future projects.

Fair warning: The internet is a big place, and you never know what kind of sites your crawler may be visiting. Filtering out sites is not feasible at the scale we are running at. Please always use proper judgement and refer to the school’s code of conduct with regards to internet safety and ettiquette.

Files and Resources

You are given a zip file containing a mostly empty .NET Core solution. Your job is to implement MyCrawler.cs. Your web crawler should write out all of its edges every hour into Data Lake Storage using date formatted folder paths. For example, if your web crawler is crawling on May 1st, 2019 at hour 14:00 (Military clock), after running for an hour it should generate a JSON file in the following form:

MyTeamName/2019/5/1/14/902d3cfe-a32c-4413-81cb-b4ab18f19288.json

Notice that we use a Guid for the file name, this is to avoid conflicts with your other web crawler programs that are running simultaneously on other App service instances.

We will be using Azure App Service Web Jobs to host our web crawler (console app program).

Please download and install the following:

You are permitted to use the following NuGet packages:

1. NewtonSoft.Json v12.0.1
2. HtmlAgilityPack v1.9.2
3. Microsoft.Azure.DataLake.Store v1.1.15

4. Microsoft.Rest.ClientRuntime.Azure.Authentication v2.3.7

What you are given:

1. A Solution file containing a console app project, and some very basic skeleton code.
2. A partner of your choosing to work on this project, and the remainder of the projects in the class.

There are many resources on the web for learning about web crawlers and data lake storage:

1. https://en.wikipedia.org/wiki/Web_crawler
2. https://cl.lingfil.uu.se/~joerg/ir14/Lecture 3 - Web Crawling 2014.pdf (Highly recommended read)
3. https://azure.microsoft.com/en-us/services/storage/data-lake-storage/

Building

From now on, you may use a single SLN for all the projects you will create for projects 1-4.

This way, you can easily make updates to older projects as we move on, and have everything in one place.

When you turn in your project, you can just zip the entire solution folder, and I will just grade the project of interest.

Deploying to Azure

You will need to setup an Azure App Service plan and configure Web Jobs to run your application.

We will be reviewing how to deploy console apps to web jobs in class, if you are not able to attend, please ask your classmates for help or post on Piazza.

Testing

While there are no “tests” for your web crawler. There are some requirements that must be met, so there is consistency between projects.

1. Your web crawler should be seeded by randomly selecting urls from the sites.txt file. (Hint: seed your links queue with more than one random url just in case the first link really doesn’t contain any other links.)
2. Your web crawler should write a json file to Data Lake Storage every hour based on the following date path format: MyTeamName/YYYY/M/D/H/{Guid}.json
3. Your JSON will be a list of Source + Destination + DateTime tuples. Your source and destination links should be in canonical

form. i.e:

www.facebook.com -> facebook.com

WWW.facebook.com -> facebook.com

https://www.facebook.com -> facebook.com

http://www.facebook.com -> facebook.com

facebook.com/ -> facebook.com

An example json file will be provided for you.

4. Your crawler should be stable and resilient to poorly formatted and malicious web pages. The web crawler must be able to run for a full hour, before dumping all of its saved memory to data lake storage, and then
   restarting anew.
5. You should be able to deploy and automatically have your web crawler be run from Azure App Service Web Jobs.
6. Your web crawler should scan the web in a BFS manner by downloading the html page for the current site, finding links on the html page, cleansing them, and then repeating on those new links.
7. Every Source + Destination pair in your json should be unique.

Grading

If you meet all of the above requirements, you will receive a grade of 100.

For every component that is missing (i.e. links are not in canonical form, output is not in date path format, output is not a json file, crawler crashes before making it to one hour, etc.) you will receive a 10 point deduction for each violation.