Introduction

The goal of this project is to provide an alternative way to access a website if your internet goes down. The program will store a backup of every site you choose to save so that you can access any site you have been to, even if your internet or the site goes down. This website can be cached by adding a browser extension that will allow you to cache it or by pasting a URL into a website. This system will allow the user to have easy access to websites that they hope to save indefinitely. This system will be integrated with the tool HTTrack and will provide added functionality by offering better data organization and ease of use for the users. The stakeholder is the TA for CSCI 150, Keyur Pawaskar. He will be acting as the company representative to help guide our team throughout the process of creating this system.

User Requirements

Features

This project will allow users to paste a website URL into an entry box on a website. They may also utilize a browser extension that has a URL entry box. As a result, users can access this website's copy which will be locally stored on the device on a local web server with a local address. Once the user pastes the URL or uses the browser extension, they can view the status of their cached website to see whether it has or has not completed downloading. This system will be able to function on Mac, Linux, and Windows machines in the browser. In addition to creating a downloaded copy through HTTrack, our team will provide added functionality through the UI, allowing users to categorize, search, sort, and display their websites.

Integration Requirements

This system will use HTTrack to store cached versions of websites. It also uses a controller program that we will design that facilitates the interaction between HTTrack and the browser. In addition, we will use MySQL to organize the data. While HTTrack does offer a GUI or graphical user interface, it is old and outdated with basic limited features that list past download files. It does not work in a browser. It also does not work on Mac or Linux. Our program will control HTTrack through the command line.

Local Installation

When installed, the program will create a local directory that stores the database of saved websites that it will use to access the websites. Since it is installed locally, it can be accessed even if the internet connection goes out. Any previously cached website can then be accessed through this.

User Interface

The user will be able to access a web interface that shows all downloaded websites in a list, in addition to the website that downloads their website with a URL entry box. The user can categorize their saved websites to allow for better organization. On top of this, they will be able

to give their cached website titles, which will enable users to be able to search their cached websites as well as sort through their saved sites through criteria such as sorting by date saved and sorted alphabetically. They may also rename their websites and delete them as needed.

Constraints

Since the website bank can only save copies of the website passed through HTTrack, it will not be able to update regularly. Therefore, this will not be useful for seeing the updates on websites and will work more as a "screenshot" of the website. Thus, certain websites that require an internet connection to be useful, such as a Desmos homework solver, will not be purposeful to save in this web bank. In addition, certain websites may not be compatible with saving. In this case, we would need to create a blacklist of sorts. There may also be some websites where saving information could become legally or ethically problematic, in which case would also be added to a blacklist. Lastly, some websites may take too much data to download, so we must specify a maximum file size.

Use Cases

Internet Outage

If the user's internet is slow or offline, they can access the cached version of the site they are trying to get to. This can provide information that you need even if they are offline.
The actors in this case would be the user themselves.

- The user sees that they are offline
- The user wants to access a website he uses frequently
- The user goes to the site in question
- The user clicks the browser extension to load the cached version
- The user can now read the cached version of the site

Intermittent Internet

If the user's internet is slow, the user can go to the site in the moments when their internet is working and store a cached copy. Depending on how deep the user decides to recursively store links, this can potentially grab the entire site. That way their browsing is not interrupted even if the website or their internet goes down again.
The actors in this case would be the user themselves.

- The user has internet that is going in and out
- The user wants to access a website he uses frequently
- The user goes to the site in question
- The user clicks the browser extension to save a cached version
- The user can now read the cached version of the site even if their internet cuts out

Server Outage
If a site a user likes is offline, they can access the cached version of the site they are trying to get to. This can provide information that they need even if the site is down.
The actors in this case would be the users themselves.

- The user sees that the site they like is offline
- The user wants to access the website anyway
- The user clicks the browser extension to load the cached version
- The user can now read the cached version of the site

Priorities
Automation of HTTrack or similar downloader
Front-end web development of URL input site or Chrome extension that takes the URL
Automation of local web server hosting from download files through node js server or similar python local web server
Sorting options for downloaded data, such as date and name
AI integration to summarize downloaded text data when questioned about it

Expected Milestones (for the prototype)
The first milestone is displaying the downloaded website data on a local web server with tools such as a node js web server or python web server.
The second milestone is automating the process of uploading new website data to the database.
The third milestone is to automate the website downloader tool (HTTrack), the download data will need to be moved into a database such as MySQL.
The fourth milestone is to integrate the downloading of files from HTTrack and move them into the location where the second milestone was achieved.
The last milestone is to have a website or web extension to pass the URL into the third milestone.

Our goal is to be able to have a working user interface that works on all systems: Linux, Mac, and Windows, and to get our system to function with two websites by mid-October. After, we plan to add more advanced functionalities and increase scope. These features include calling on an LLM (large language model) to interpret the collected data to more easily find the information the user is looking for.