

Search Summarizer

CSE2023 – Robotic Process Automation

Review - 3

by

Harisha S - 20BRS1127

Sachit Nair - 20BRS1140

Alan Shaji - 20BRS1024

Dinesh D - 20BRS1015

Abstract:

As engineering students, we are accustomed to having to gather information on a particular topic from various resources online. However, a major pain point in this procedure is the fact that most of the time no single website contains exactly what we require, information is often scattered and not exactly to the point. This means that we have to manually jump from website to website and check if it contains the information we are looking for, this is highly inefficient and could be optimized with the help of automation and AI. Our project basically boils down to alleviating this manual effort and providing the user with accurate results gathered from the multiple resources in a single document. The automation aspect of this project comes from the data-gathering phase. We have to collect the websites that come up when the user's search string is entered into a search engine like Google. For the best possible results, we would have to search the content of every website that comes up, however, this is infeasible due to the sheer number of websites, hence we would set a limit on the number of websites referred. So we would keep an upper limit to the number of pages that our process would go through as the websites that Google provides us, come sorted according to maximum relevance. We would then scrape the contents off these websites using automation software like UIPath. We are going to use automation software to open each website found by Google when we enter the keywords. Then the software extracts the text on the website (either all the text or a part of it which is relevant to us and has the keywords in it). Then we would analyze the text using Machine Learning techniques to understand and see how relevant the information is to the user according to the keywords he has used in the search engine. Then we will put together all the information we have gathered using AI so that they flow together in a way that the user understands. Thus, we automate the data collection process using software bots.

Introduction:

The proposed system provides a quick and efficient way to gather information about a particular search query using RPA tools (UiPath, Orchestrator) and Machine learning (Text summarizing model). The user is presented with a frontend containing a search bar where the query may be entered, and the summarized result of contents from various resources are presented as the output.

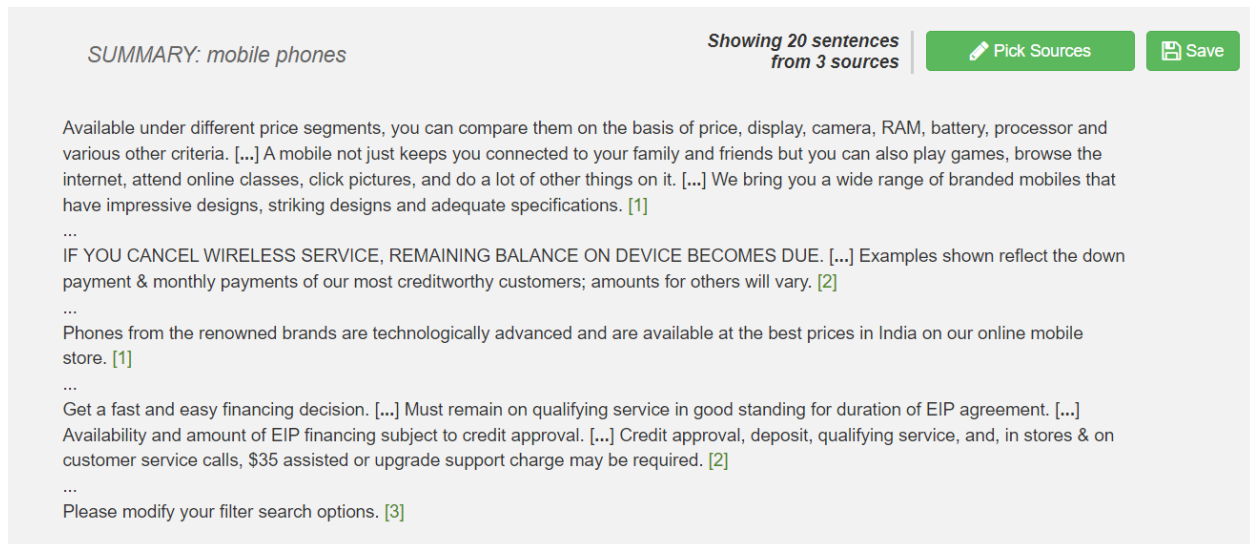
The frontend is a web application written using HTML, CSS and Javascript. Orchestrator is used as a backend to run the automation on a local machine which acts as a server. The results are then passed through a machine learning model that extracts the important information from the text, this result is then displayed on the website for the user.

Existing System:

Our topic is a search summarizer which takes the user's query and searches multiple pages for the results and summarizes the consolidated data.

SenseBot is a web service which has the exact features of the robot which we wish to make. The summary Sensebot generates is irrelevant and it scrapes data from websites which don't hold a lot of content. This makes the user experience really poor and doesn't add much novelty to it.

An example of SenseBot queries can be seen below:



The screenshot shows a web interface with a summary for the query "mobile phones". At the top, it says "SUMMARY: mobile phones" and "Showing 20 sentences from 3 sources". There are two buttons: "Pick Sources" and "Save". The summary text is as follows:

Available under different price segments, you can compare them on the basis of price, display, camera, RAM, battery, processor and various other criteria. [...] A mobile not just keeps you connected to your family and friends but you can also play games, browse the internet, attend online classes, click pictures, and do a lot of other things on it. [...] We bring you a wide range of branded mobiles that have impressive designs, striking designs and adequate specifications. [1]

...

IF YOU CANCEL WIRELESS SERVICE, REMAINING BALANCE ON DEVICE BECOMES DUE. [...] Examples shown reflect the down payment & monthly payments of our most creditworthy customers; amounts for others will vary. [2]

...

Phones from the renowned brands are technologically advanced and are available at the best prices in India on our online mobile store. [1]

...

Get a fast and easy financing decision. [...] Must remain on qualifying service in good standing for duration of EIP agreement. [...] Availability and amount of EIP financing subject to credit approval. [...] Credit approval, deposit, qualifying service, and, in stores & on customer service calls, \$35 assisted or upgrade support charge may be required. [2]

...

Please modify your filter search options. [3]

Fig 1: The summary for "Mobile phones" it has generated

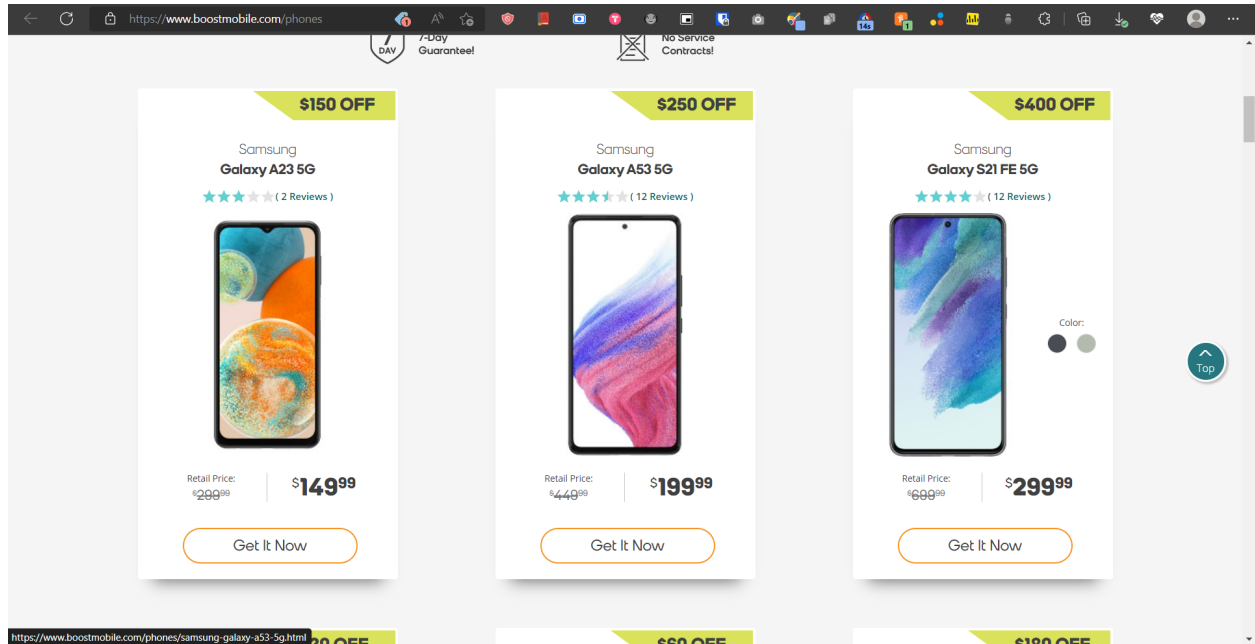


Fig 2 : Website Source 1 taken for content by SenseBot. Doesn't contain much textual data

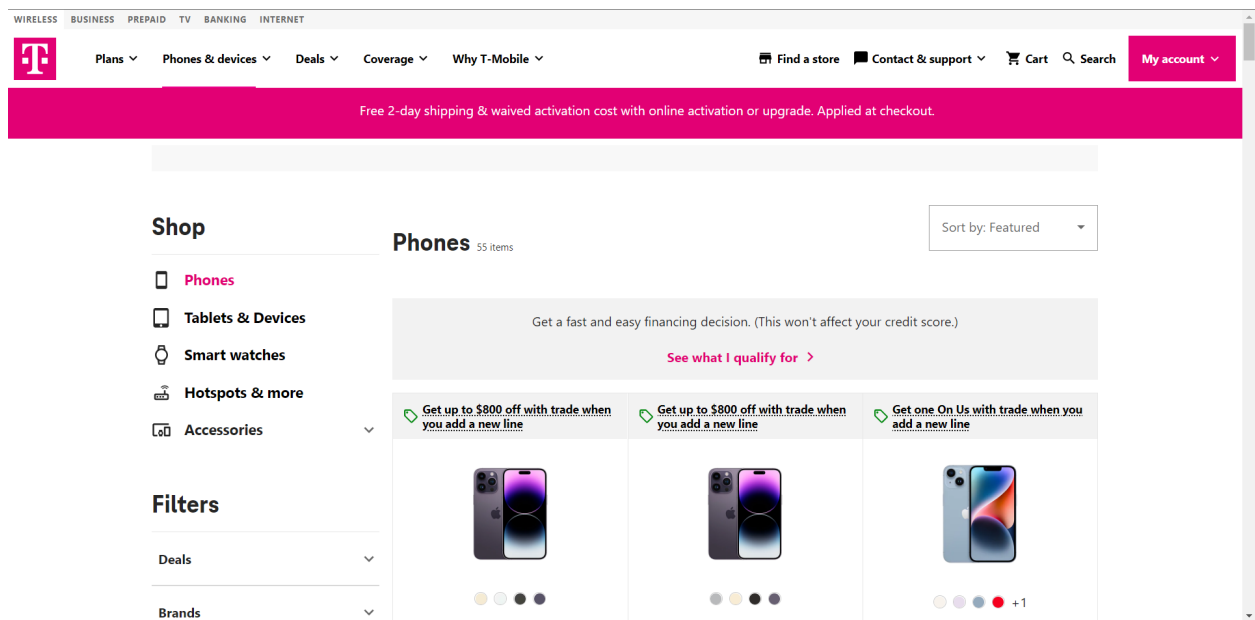


Fig 3 : Website Source 2 taken by SenseBot. It doesn't contain much textual data

Excluding online services, there are also research papers on Search Summarization, although only a few are worked on it in recent years. In these methods, the HTML file is converted to an object of the DOMDocument class. With the use of the DOMXPath, the generated object provides a fast and convenient interface to search for the contents of the document tags. The result of this block of operations is then to convert the user's request into a set of documents whose content intersects with the request text.

Proposed System:

Our proposed system works by utilizing a browser feature as its core. Every browser has a read mode that strips off unnecessary data from a website including ads, navigation bars, table of content, featured articles page, etc. By enabling read mode in every website we are visiting, we are able to determine whether or not a page is content rich. If the page is content-rich, read mode is enabled and the text is condensed into a definite structure. This structure is uniform regardless of what the source website is. Using this knowledge, we scrape the content from multiple websites, store it and then pass it to summarizers. This mode of workflow ensures that irrelevant websites are not scraped for data, and only content-rich websites are utilised. As we are implementing our automation using UiPath we have settled on the current proposal. Future scope on this aspect will be to develop our own HTML parser which extracts the body text regardless of whichever website it is.

Background:

a) Methodology

We will get the query from the user and enter it in the google search engine. We select the first ten links and open all those links and search for text scraping. If the text could be scraped then we scrape it or else we move it to the next link. Here we restrict ourselves to two links and extract the data from those links. Then we will summarize this text and display it on the website.

b) Design - System Architecture of the project

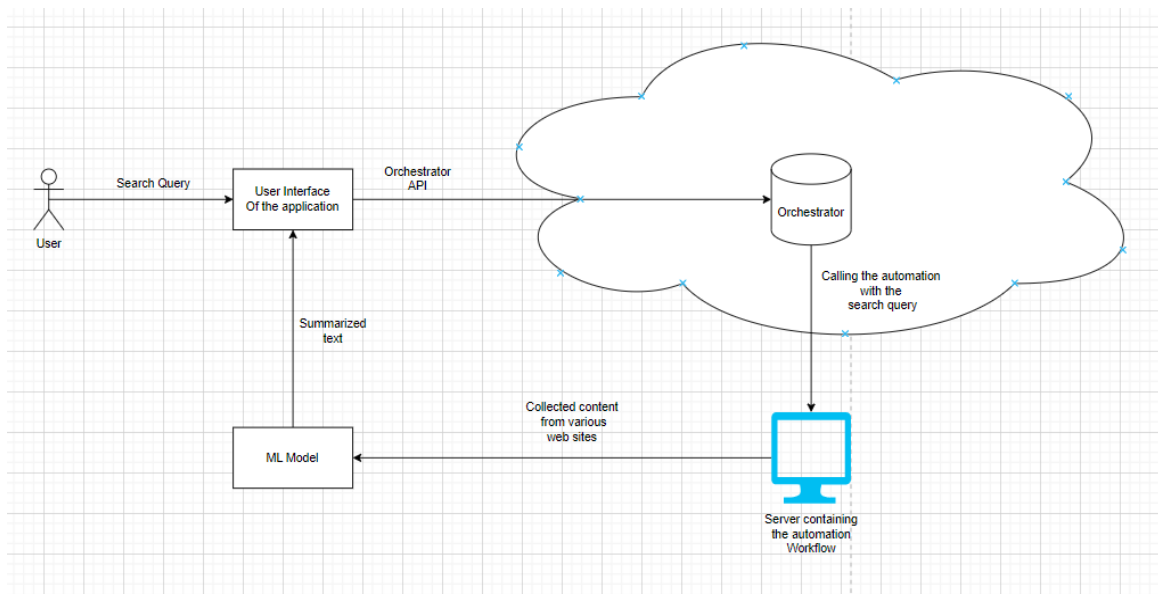


Fig 4 : System Architecture

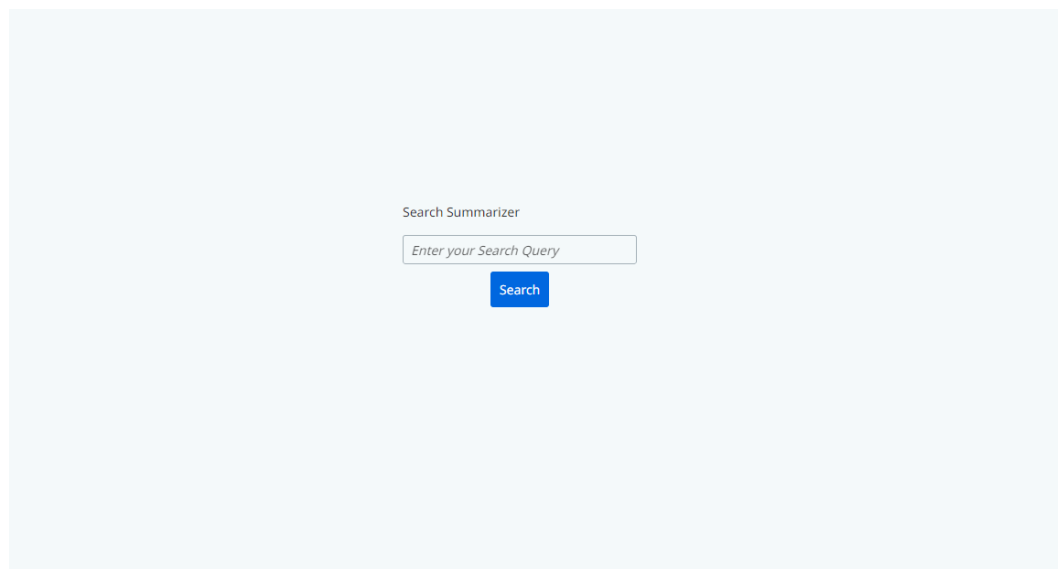
The basic flow of the project is as follows:

1. The user enters a search query in the front end of the application.
2. The search query is sent to the backend as an argument to the workflow.
3. The workflow returns a text file containing the content from the first few websites that come up when the search query is typed into google (All this is achieved using the process on UiPath, running on the server machine).
4. The text returned is passed into a summarizing model that extracts the essence of all the collected texts from various websites.
5. The summarized information is presented to the user via the frontend.

Thus, the user doesn't have to manually go through different websites and gets the essential information without any hassle.

Modules

a) User Interface

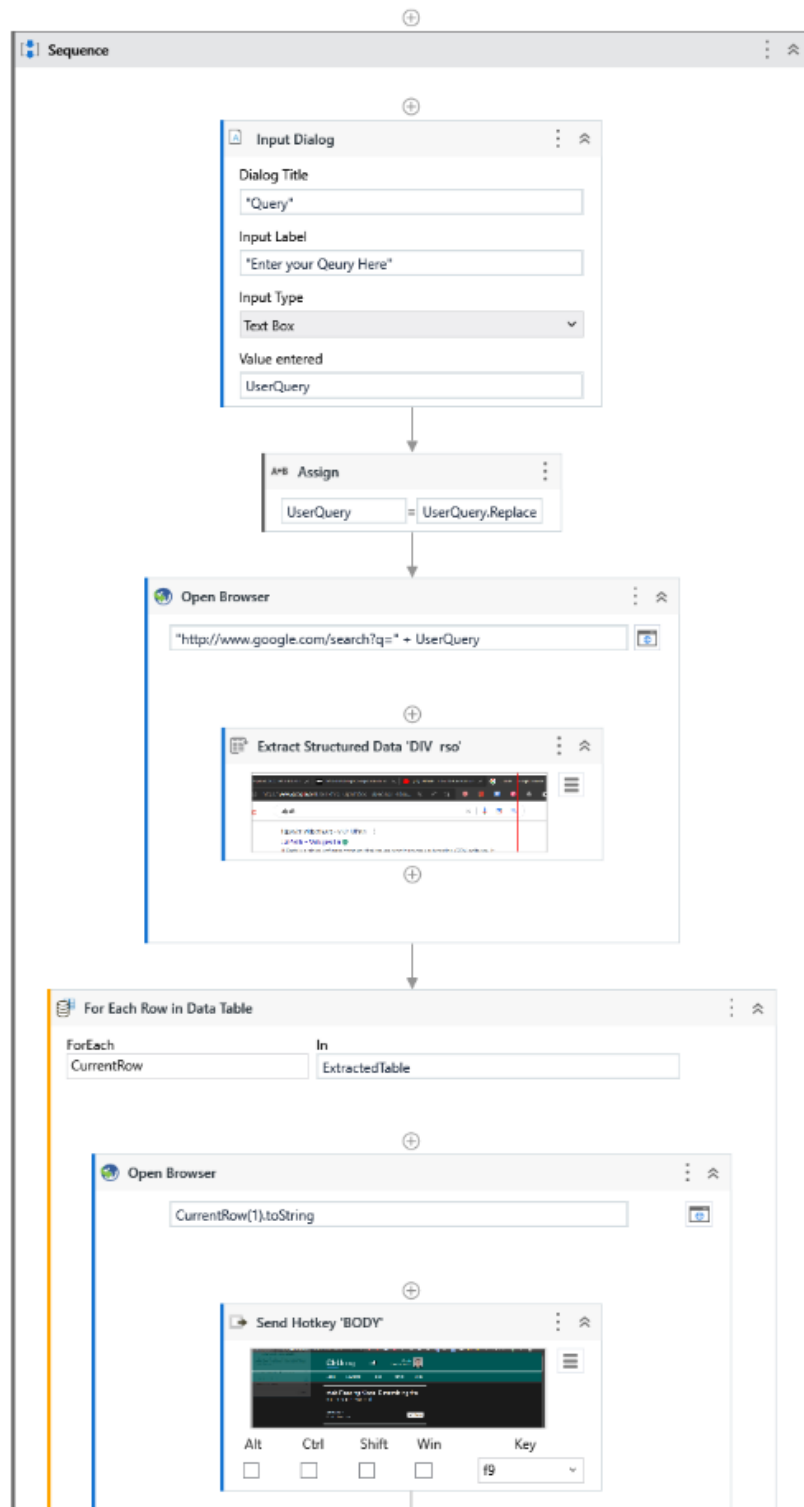


b) Web Scraping Module

The UiPath process acts as one of the three main components of the project. The process is stored on the Orchestrator cloud. When the user inputs a search query, our website forwards it to the Orchestrator and feeds the search query into our UiPath process.

The process:

1. The process starts with the search query being fed into it using Input Dialog activity
2. We then swap all the spaces between the words with '+' to issue a direct search query for the given topic. This will be in turn appended with the google url to produce a search output. The result is stored in a variable called UserQuery
3. An 'Open Browser' activity is utilized to open the Edge browser and type in open 'www.google.com' + UserQuery and write the search query into the search box.
4. Once the query is executed, the Data Scraping module is used to extract structured data from the google results page. It extracts the first 10 links to get the required information for compiling information.
5. Out of the 10 links collected, only 2 links are extracted for text. We have a backup of 8 links because not all websites contain useful text-based, article-like information. In case a website doesn't have useful information, it is skipped. If relevant, then the text is scraped and stored
6. Data is scraped from the websites by entering into the Immersive Reader model which removes filler content, ads, and UI Elements and formats the resultant crux into a uniform web format



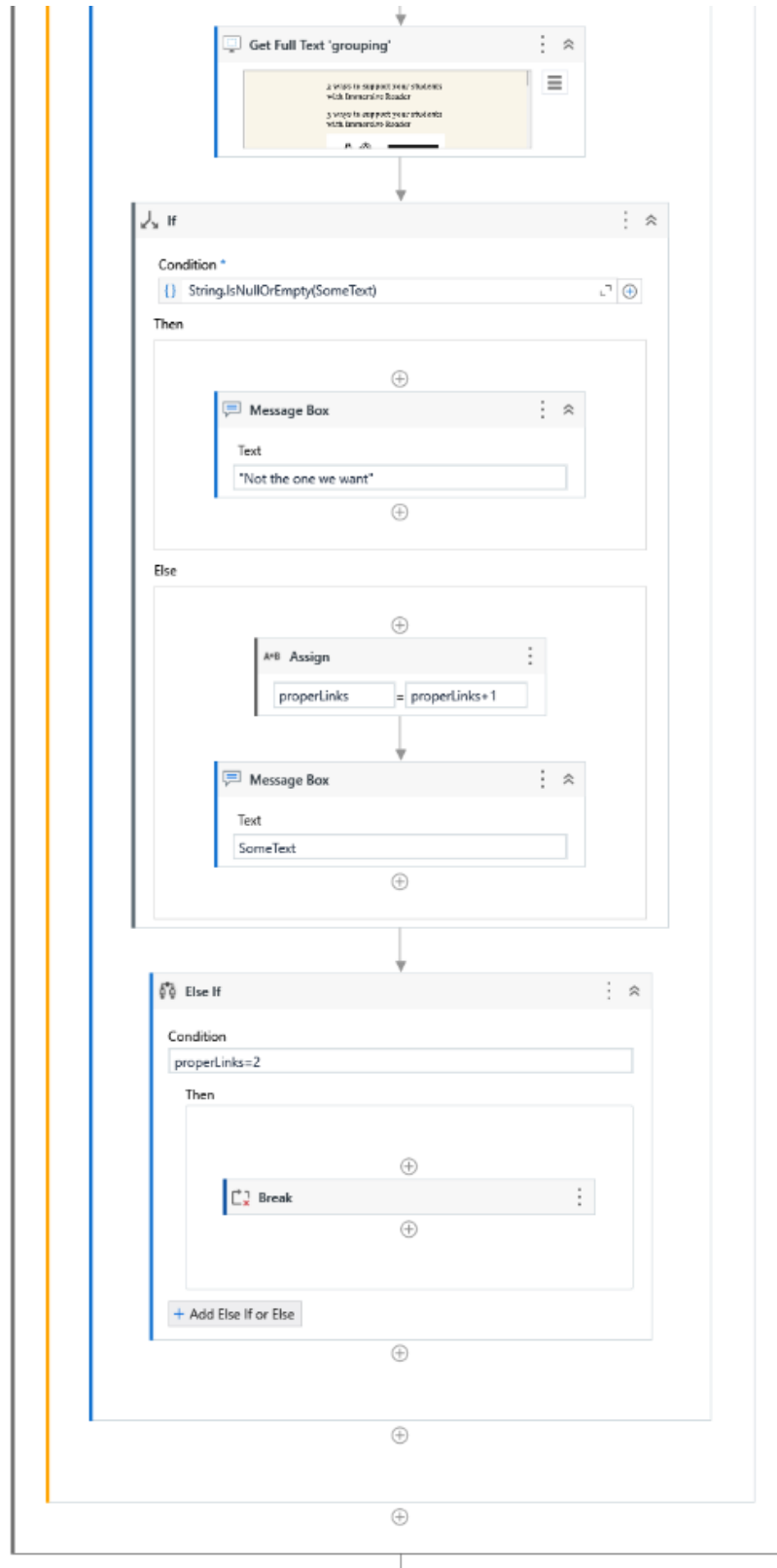


Fig 5: UIPath Workflow of Text Extraction Module

c) Summarizing Module

Once the data is extracted from the links the data is given to the summarizer module. This summarizer module consists of ML Skill Activity. In order to use the ML Skill Activity we deployed the ML Package available in the Out of box ML Packages in UiPath Cloud. We made use of the ML Package of Text Comprehension i.e. Text Summarization Package. We deployed that package and used that ML Skill in our activity to summarize the extracted text.

TextSummarizationPackage						
Upload new version						
VERSION	PIPELINE RUNS	ML LOGS				
Package Version	Created Time	Change Log	Status	Training Enabled	Recommend GPU	
1.0	2022-11-04 09:42 pm		Deployed	×	✓	⋮

1 - 1 of 1 Page 1 / 1 Show items: 5

Fig 6 : Deploying TextSummarizationPackage (ML Package)

ML Skills				
Create new				
Search...				
ML Skill Name	Package Name	Status	Deployed	Prediction
TextSummarizationSkill	TextSummarizationPackage	Available	2022-11-20 12:13 am	3

1 - 1 of 1 Page 1 / 1 Show items: 5

Fig 7 : Deploying the TextSummarizationSkill (ML Skill)

TextSummarizationSkill		Stop	Rollback	Modify current deployment
Status	Available	GPU	×	
ML Package	TextSummarizationPackage	Version	1.0	
Prediction	3	Deployed	2022-11-20 12:13 am	
Modified on	2022-11-21 10:45 pm	Inactivity Period	30 Days	
API Key	51b794a7-0458-4883-9145-c92ceb2c8609			
Url	https://ai-upath.deskover.com/public/mlskills/d2b02125-c...	Input Type	Json	
Input description	Text to be summarized as a string. On small inputs (less than 300 characters), the model will just return the unaltered input			
Output description	Summarized text as a string.			
Description				

VERSION	ML LOGS	STREAMING LOGS
Package Version	Created Time	Change Log
1.0	2022-11-04 09:42 pm	

Fig 8 : Deployed TextSummarizationSkill

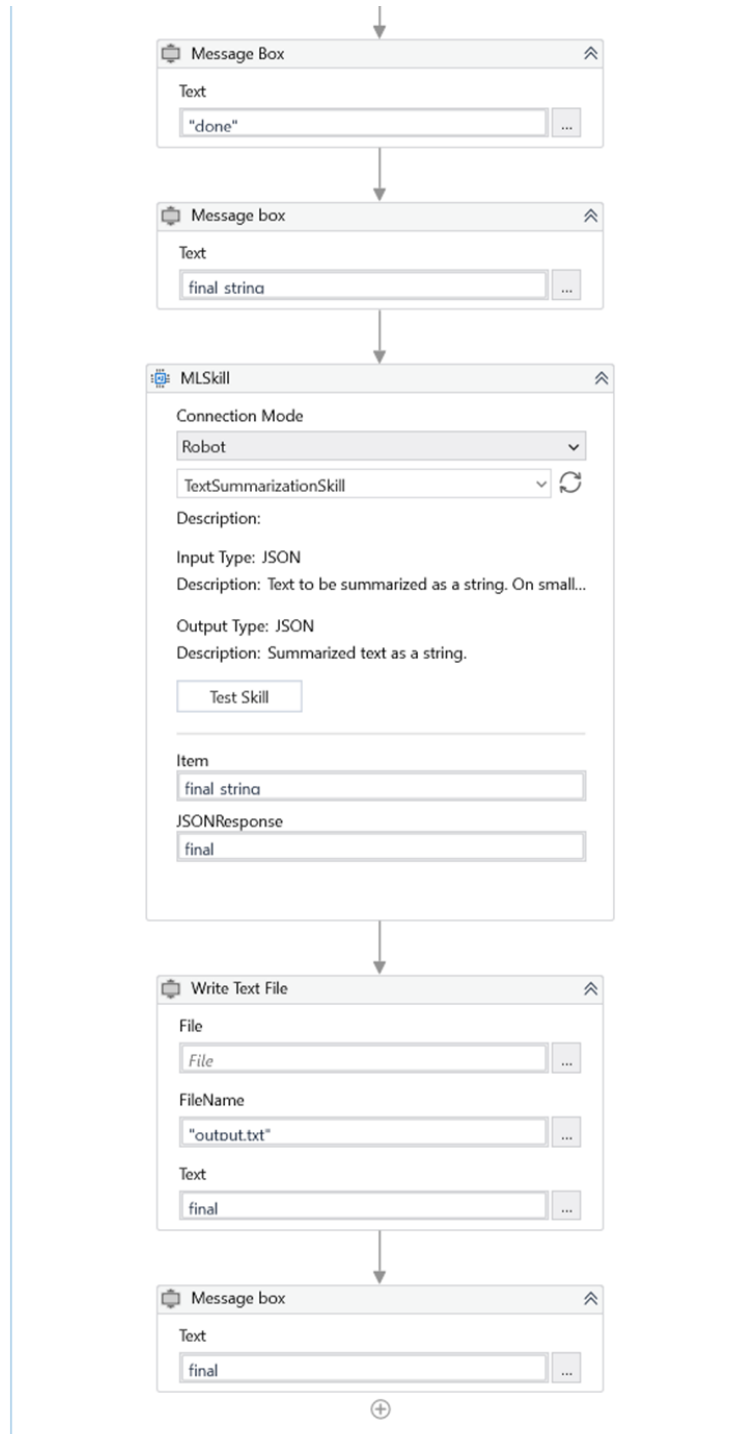


Fig 9 : UIPath Workflow of Text Summarization Module

In UIPath, we added the ML Skill Activity and gave the connection mode as Robot. Under that gave the TextSummarizationSkill which was deployed in our Orchestrator Account. Then we gave the extracted text as input to this ML Skill Activity. Then the output of this ML Skill Activity is stored in a Text File named 'output.txt' using Write Text File Activity. Also displayed the summarized text in a Message Box.

d) API Module

Orchestrator API:

When the building of the process is complete on UiPath Studio, we have to use the Orchestrator API to trigger the process from our website remotely. First, we create a tenant in the Orchestrator. Then, we create a folder which will contain our process that we will publish later. After creating the folder and assigning the necessary roles to the user, we create an attended robot which will run our process from the Orchestrator. We then create a machine and assign the user of the folder to the machine so that the user can control the machine to control the robot and run the process.

Then we open UiPath Studio and publish the process to Orchestrator Tenant Process Feed. In the folder that we created in Orchestrator, create a new process and select the package name that we gave while publishing the process. After creating the process, we can run the process from our Orchestrator.

Orchestrator App:

Since we require funding to host our website, we used Orchestrator Apps to create a basic user interface to run the process.

We enter the search query in the search box and click search. We have linked the search button to trigger the process in the Orchestrator and when we hit the search button, the process starts to run.

Related Problems

The main problem tackled in our project is getting information from the web and summarizing it for the user. The aim of the project is to save the user time and energy from having to manually search various resources by themselves. This aim may be applied to many other related problems as well.

1. Book summarizer: Reading is an activity that is extremely beneficial but unfortunately it also takes a lot of time. In the busy lives we lead, reading entire books to get the knowledge they hold is very difficult. Thus a project could be developed with the aim of summarizing books for users, providing them with the short and sweet version of all the condensed material.
2. Application that tracks stock prices: Since a component of our project is basically gathering data from a web browser, the same concept may be applied to extract data from stock exchanges. This information can then be presented in a graphical manner for the user to track the rise and fall of stocks of interest.

Result Analysis

The project has been completed as planned, the main aim of this project, that is, to get a search query from the user and to return a summarized and condensed version of all the information they would have found if they had gone through the top websites suggested by google manually has been completed.

The summarizing is handled by using built in machine learning skill by UiPath, thus it is also very accurate.

We have tested out the project on various search queries and the output is satisfactory.

Conclusion

Our project optimizes the search results from Google for a user who is looking for information about a topic. This is accomplished through web scraping relevant search results and summarizing them entirely using UiPath Activities. This reduces the amount of time spent by a user searching for relevant information and gives back a concise summary along with the references. Innovation in web scraping can result in more precise scraped results over dynamic web pages and images. The product can also be enhanced by utilizing state-of-the-art Natural Processing models which understand and generates new, precise summaries. It can also be extended to include relevant images in correlation with the summary that is being presented.