**Proposer Details**

| Group Number | *G16* |
| --- | --- |
| Registration Number of Group Members | *2020\_CS\_145*  *2020\_CS\_148* |

**Proposal Details**

|  |  |
| --- | --- |
| ***Project*** |  |
| Proposed Project Title | *Fashion Clothes Info* |
| Executive Summary | *In search of good quality and affordable clothes my time was wasted in exploring so many websites of different brands. I tried to find out when the best time to buy clothes but I did not find any efficient way. So, I decided to build a small program to automatically collect the data from the website of my choice. It extracts information for my specific interest and displays the category, price, colors, and many more features of clothes I want to buy that I was searching. So, basically, Web Scraping is a technique used to extract data from websites through an automated process. Our project is about scrapping data of clothes with their attributes from websites and then putting them in a way that the data is sorted in well formatted form using different technique of Data Structures. In this project we use algorithms for data management. We are formulating the ideas of different algorithms and sorting techniques. All these techniques of algorithms are implemented on our data. For example: Types of sorting like merge sort, Insertion sort, Bubble sort, selection Sort and many more concepts we studied in DSA. With the help of all these techniques, we sort the attributes (data) in ascending and descending order up to user’s choice. We try to illustrate a progress bar that will display clearly that how much data has been scraped and if we want to pause or resume the scraping. Basically we are using different algorithms techniques and at the end we will find the time complexity taken by the whole process. To make UI interactive it is necessary to display data in such a format that is user friendly. Here, a user can see/buy any type of clothing product rather it’s for men, women, kids, bridal, winter summer, new in or all, etc.* |
| ***Business Case*** |  |
| Outline the business need for the project | *Many of the clothing brands and websites have an easy access in searching the clothes with their brands and related prices according to latest fashion. This provides a benefit to clothing brands to save their time and increase their rating.* |
| End user of the product | *Our target is those Customers that wants to buy products using online resources.* |
| Motivation for Project | *We can collect data at greater volume than a single human could ever hope to achieve.* |
| State the level of impact expected should the project proceed and implications of not proceeding | *We can access and collect any data that is available on their website that we scrapped. To store data in efficient way and to save people’s time, this technique is fruitful in many ways.* |
| ***Technical Details*** |  |
| Name of Entity | *Clothes* |
| Attributes of Entity  (Minimum seven attributes/rows can be increased) | |  |  |  | | --- | --- | --- | | *Name* | *Data Type* | *Description* | | *Clothes’ ID* | *Integer* | *Brand no. of clothes* | | *Category* | *String* | *Category of clothes like men, women, kids* | | *Collection* | *String* | *Collection of clothes i.e. summer, winter, new arrival.* | | *Brand* | *String* | *Name of brand* | | *Size* | *String* | *Size of clothes like Small, Medium, Large* | | *Combination* | *String* | *Combination of Clothes i.e.1pc, 2pc, 3pc* | | *Price* | *String* | *Different prices of clothes from low to high.* | |  |  |  | |
| Sample of Scrapping Source |  |
| Github Repository Link | *https://github.com/Aiemenaltaf/CS261F21PID56* |
| Sorting Algorithms | *Insertion Sort, Selection Sort, Bubble Sort, Merge Sort, Quick Sort, Counting Sort, Bucket Sort, Radix Sort, Heap Sort, Tree Sort, Cycle Sort, Shell Sort, Comb Sort, Sleep Sort* |
| |  |  | | --- | --- | | **Algorithm Name** | **Description(Each algorithm in 2-3 lines)** | | Insertion Sort | *Insertion sort is a technique used like when we play cards. The array is virtually split into a sorted and an unsorted part. Values from the unsorted part are picked and placed at the correct position in the sorted part.* | | Selection Sort | *The selection sort algorithm sorts an array by repeatedly finding the minimum element from unsorted part and putting it at the beginning.* | | Bubble Sort | *Bubble Sort is the sorting algorithm that works by repeatedly swapping the adjacent elements if they are in wrong order. Here, algorithm compares the first two elements, and then swaps.* | | Merge Sort | *Merge Sort is a Divide and Conquer algorithm. It divides the input array into two halves, calls itself for the two halves, and then merges the two sorted halves.* | | Quick Sort | *Quick Sort is a Divide and Conquer algorithm. It picks an element as pivot and partitions the given array around the picked pivot.* | | Counting Sort | *Counting is a sorting technique based on keys between a specific range. It works by counting the number of objects having distinct key values. Then doing some arithmetic to calculate the position of each object in the output sequence.* | | Bucket Sort | *Bucket sort is mainly useful when input is uniformly distributed over a range.* | | Radix Sort | *Radix Sort is to do digit by digit sort starting from least significant digit to most significant digit.* | | Heap Sort | *Heap sort is a comparison-based sorting technique based on Binary Heap data structure. We first find the minimum element and place the minimum element at the beginning. We repeat the same process for the remaining elements.* | | Tree Short | *Tree sort is a sorting algorithm that is based on Binary Search data structure. It first creates a binary search tree from the elements of the input list or array and then performs an in-order traversal on the created binary search tree to get the elements in sorted order.* | | Cycle Short | *Cycle sort is a comparison sort that is theoretically optimal in terms of the total number of writes to the original array.* | | Shell Sort | *In Shell Sort, we make the array h-sorted for a large value of h. We keep reducing the value of h until it becomes 1. An array is said to be h-sorted if all sublists of every h’th element is sorted.* | | Comb Sort | *Comb Sort removes more than one inversion counts with one swap.* | | |
| Searching Algorithms | ***Linear search:***  *Linear search is a search that finds an element in the list by searching the element sequentially until the element is found in the list.*  ***Binary search****:*  *Binary search is a search that finds the middle element in the list recursively until the middle element is matched with a searched element.* |
| Searching Filters for each data type | *1 .GUI should have the option for searching based on each column.*  *2. User has choice to choose any algorithm for sorting for a particular column.*  *3. Advanced filters for string columns are used such as contains, end with, starts with etc.* |
| Multi-Level Sorting | *GUI should have the option for sorting of each column. So user have option to sort data using multiple columns.* |
| Any other features | *Scrapping task has the option to pause, start, resume and stop.* |
| ***Interfaces for your project*** |  |
| *[Draw layouts in the pencil tool. For each picture of the UI, provide the following table.]*         |  |  |  | | --- | --- | --- | | UI Component Name | Type of UI component | Purpose of UI Component/Other details | | *Sorting Algorithms* | *Button* | *To select those algorithms which we want to implement* | | *Algorithms* | *Drop-down list* | *List of Algorithms to select one Algorithm.* | | *Search* | *Button* | *To search specific algorithm.* | | *Side bar* | *Scroll bar* | *Used to scroll the screen* | | *Data Scrapped* | *Button* | *To show how much data has been scrapped* | | *bar* | *Progress bar* | *To show how much data has been scrapped* | | Clothes’ ID | *Button* | *To show list of Clothing Id’s* | | Category | *Button* | *To show list of Clothing Category* | | Collection | *Button* | *To show list of Clothing Collection* | | Brand | *Button* | *To show list of Clothing Brand* | | Size | *Button* | *To show list of Clothing Size* | | Combination | *Button* | *To show list of Clothing Combination* | | Price | *Button* | *To show list of Clothing Price* | | Time Complexity | *Button* | *To show how much time complexity is used by each algorithm.* | | |