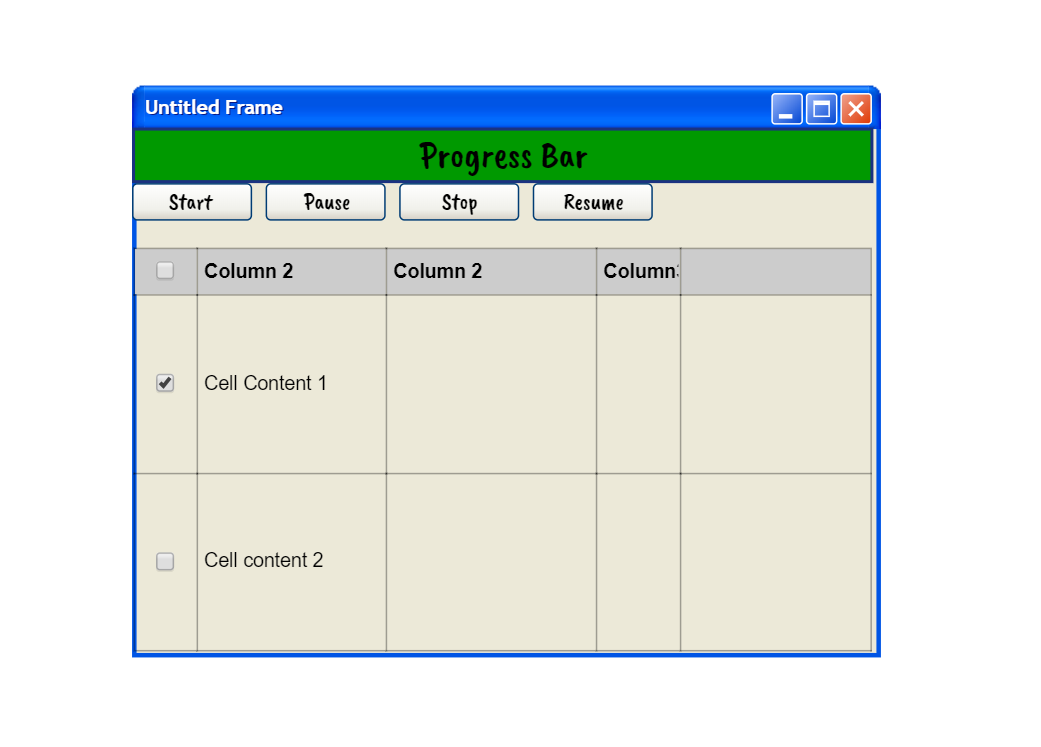
**Proposer Details**

| Group Number | *17* |
| --- | --- |
| Registration Number of Group Members | 2020-CS-134  2020-Cs-157 |

**Proposal Details**

|  |  |
| --- | --- |
| ***Project*** |  |
| Proposed Project Title | Pharmacy Scrapping |
| Executive Summary | In this project we will scrap medicine or pharma data from different websites.  **Purpose:**  Purpose is to automate data, record of sales and record of medicines. With a pharmacy scraping, you can get the medicine you need is very useful. Web-scraping represents an efficient tool to obtain data from hard-to-reach places. I am looking for a faster way to build up product cata-log.  This system is quite user-friendly. All the updates regarding the new drugs introduced in the market and other drugs related information can be found in the system. If I will scrap data from multiple sites then there is a chance that multiple items of same type can be scraped.  **Detail:**  We can sort data of a single column. There is a progress bar which will show the amount of data which is scraped. We can start, stop, Pause, Resume the scrapping. The user can choose specific algorithm from the given options. Data can be scraped up to 1 million entries. Attributes of entity will be at least 7. After sorting of column we can display time in milliseconds. Advanced filters for string columns will be implemented such as contains, end with, starts with etc. User can sort data of multiple columns at a time. User can search the scraped data for specific entry. One page of UI will display the list of chosen entity. Drop down box from which user will select the algorithm from which he/she want to work. One can find drugs by ailments, class, companies, and brands. From this project one can easily check name, brand, company, dose size and price of drug easily and in sorted manner.  **Library:**  Selenium is a web-scraping library in python that works by setting up an automated google chrome page. We can then control this by programming to do our searching for us. |
| ***Business Case*** |  |
| Outline the business need for the project | The Internet is a data store of world's information - be it text, media or data in any other format. Every web page display data in one form or the other. Access to this data is crucial for the success of most businesses in the modern world. Unfortunately, most of this data is not open. Most medicine/Pharma websites do not provide the option to save the data which they display to your local storage, or to your own website. |
| End user of the product | Actually when we are searching for a specific medicine we need to search it by typing its name but by scraping and sorting one can search specific medicine by many means i.e Name, Price, Formula, Dose size and Company name. |
| Motivation for Project | From pharmacy scraping it will be easy to automate the manual process. |
| State the level of impact expected should the project proceed and implications of not proceeding | I think this project will be of great use for many of the patients who used to buy medicines online. From this project they can search their specific medicine by many searching ways. |
| ***Technical Details*** |  |
| Name of Entity | * Syrups * Tablets * Syringes * Capsules * Supplements * Ointments * Medical Equipment |
| Attributes of Entity  (Minimum seven attributes/rows can be increased) | |  |  |  | | --- | --- | --- | | *Name* | *Data Type* | *Description* | | Name | String | User can search a specific medicine by typing its first, last or any alphabet used in that medicine. List will be shown of matching medicines. | | Company Name | String | User can search for specific medicine easily by typing company name. | | Price | Integer | User can easily check the price of specific medicine. | | Formula | String | User can search medicine by formula of that medicine. | | Dose size | Integer | User can easily check the dose size of selected medicine. | | Categories | String | User can search specific medicine from the categories of medicines. i.e Cough, fever and Headache etc. | | Type | String | User can check the type of medicine. i.e Liquid, Powder and Gel etc. | |  |  |  | |
| Sample of Scrapping Source | ***Dawaii.pk:***    ***Sehat.pk:*** |
| Github Repository Link | https://github.com/zhassan9246/CS261F21PID17.git |
| Sorting Algorithms | There will be 13 algorithms we will use for sorting whose names are given below: |
| |  |  | | --- | --- | | **Algorithm Name** | **Description(Each algorithm in 2-3 lines)** | | Selection Sort | In selection sort we start by finding the minimum value in a given list and move it to a sorted list. Then we repeat the process for each of the remaining elements in the unsorted list. | | Insertion Sort | Insertion sort involves finding the right place for a given element in a sorted list. So, in beginning we compare the first two elements and sort them by comparing them. Then we pick the third element and find its proper position among the previous two sorted elements. This way we gradually go on adding more elements to the already sorted list by putting them in their proper position. | | Merge Sort | This is a divide and conquer algorithm. In this algorithm we split a list in half and keeps splitting the list by 2 until it only has single element. Then we merge the sorted list. We keep doing this until we get a sorted list with all the elements of the unsorted input list. | | Bubble Sort | Simplest sorting algorithm. Iterates over the list, in each iteration it compares elements in pairs and keeps swapping them such that the larger element is moved towards the end of the list. | | Hybrid Sort | Hybrid Sort is a [hybrid](https://en.wikipedia.org/wiki/Hybrid_algorithm) [stable](https://en.wikipedia.org/wiki/Category:Stable_sorts) [sorting algorithm](https://en.wikipedia.org/wiki/Sorting_algorithm), derived from [merge sort](https://en.wikipedia.org/wiki/Merge_sort) and [insertion sort](https://en.wikipedia.org/wiki/Insertion_sort), designed to perform well on many kinds of real-world data. | | Quick Sort | In this algorithm we partition the list around a pivot element, sorting values around the pivot. In my solution I used the last element from the list as pivot value. Best performance is achieved when the pivot value splits the list in two almost equal halves. | | K-Select/Quick Select | Quick-select/K-Select is a selection algorithm to find the Kth smallest element in an unordered list. It is related to the quick sort sorting algorithm. | | Counting Sort | This algorithm does not do comparison between the elements. We use the mathematical properties of the integers to sort. We count how many time a number has come and store the count in the array where index is mapped to key’s value. | | Heap Sort | We create two segments of the list one sorted and one unsorted. In this we use heap data structure to efficiently get the max element from the unsorted segment of the list. Heap method uses recursion to get the max element at the top. | | Bucket Sort | Bucket Sort is a sorting algorithm that divides the unsorted array elements into several groups called buckets. Each bucket is then sorted by using any of the suitable [sorting algorithms](https://www.programiz.com/dsa/sorting-algorithm) or recursively applying the same bucket algorithm. | | Shell Sort | Shell Sort involves sorting elements which are away from each other. We sort a large sub-list of a given list and go on reducing the size of the list until all elements are sorted. | | Radix sort | Radix sort is one of the sorting algorithms used to sort a list of integer numbers in order. In radix sort algorithm, a list of integer numbers will be sorted based on the digits of individual numbers. Sorting is performed from least significant digit to the most significant digit. | | Tree Sort | Tree sort is a sorting algorithm that is based on [Binary Search Tree](https://www.geeksforgeeks.org/binary-search-tree-set-1-search-and-insertion/) 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. | | |
| Searching Algorithms | **Linear Search:** A linear search or sequential search is a method for finding an element within a list. It sequentially checks each element of the list until a match is found or the whole list has been searched.  **Binary Search:** Binary search is an efficient algorithm for finding an item from a sorted list of items. It works by repeatedly dividing in half the portion of the list that could contain the item, until you've narrowed down the possible locations to just one.  **Exponential Search:** This mechanism is used to find the range where the search key may present. This mechanism is used for infinite number of entries.  **Fibonacci Search:** Fibonacci Search is a comparison-based technique that uses Fibonacci numbers to search an element in a sorted array. |
| Searching Filters for each data type | **By Name of Medicine:** You can search a specific medicine by typing its first, last or any alphabet used in that medicine. List will be shown of matching medicines.  **By Company Name:** You can also search by the company name.  **By Formula:** You can also search by formula of medicine. |
| Multi-Level Sorting | Use a lambda function with a tuple to sort with two keys. |
| Any other features | **Save Data:**  User can save data in his/her pc for future use. |
| ***Interfaces for your project*** |  |
| |  |  |  | | --- | --- | --- | | UI Component Name | Type of UI component | Purpose of UI Component/Other details | | Label | Label | To ask user to write any word to sort or select the algorithm. | | Combo Box | Combo Box | It is a drop down box from which user will select the algorithm from which he/she want to work. | | Buttons | Button | There are two buttons on this UI one is cancel which is used to cancel the sorting and other is next and this helps to move to next UI page. | | Window Frame | Frame | Used here to enhance the quality of UI. | |  |  |  | | |



|  |  |  |
| --- | --- | --- |
| UI Component Name | Type of UI component | Purpose of UI Component/Other details |
| Progress Bar | Progress Bar | Progress bar showing the progress of tasks/ number of entities scrapped. |
| Table | Table | Table will shoe the entities which are sorted or to be sorted. |
| Buttons | Button | There are two buttons on this UI one is cancel which is used to cancel the sorting and other is next and this helps to move to next UI page. |
| Window Frame | Frame | Used here to enhance the quality of UI. |