Central Real Estate Scrapper



Project Supervisor

Mr. Samyan Qayyum Wahla

Project Members (Group # 21)

|  |  |
| --- | --- |
| Haider Ali | 2020-CS-137 |
| Ali Tariq | 2020-CS-142 |

Department of Computer Science

University of Engineering and Technology, Lahore

Pakistan

**Table of Contents**

Proposal Statement……………………………………………………………………………………………..… 3 Business Use………………………………………………………………………………………………………4

1. Business use
2. Motivation of Project
3. Level of Impact
4. End user of the product

Technical Details………………………...………………………………………….……………………………4

1. Entities…………………………………………………………………………………………………………………………..5
2. Sorting Algorithms…………………………………………………………………………………………………………6
3. Searching Algorithms…………………………………………………………………………………………………….7

Extra Features Details…………..……………………………………………………………………………..… 8

User Interface …………………..……………………………………………………………………………..… 9

**Proposer Details**

| Group Number | *G-21* |
| --- | --- |
| Registration Number of Group Members | 2020-CS-137  2020-CS-142 |

**Proposal Details**

|  |  |
| --- | --- |
| ***Project*** |  |
| Proposed Project Title | Central Real Estate |
| Executive Summary | *Real estate advisors are facing the problems, so the solution is web scraping. So, a software will be designed in which the data will be scraped and in order to make it more convenient for the user to interact with it, there will also be a feature of sorting in which various sorting algorithms will be used. For example, users will be able to sort the data with the filters such as with entities like location, title, area and also with the price.*  *Moreover, to make it more efficient, a proper graphical user interface (GUI) will be designed in which the user will have the following features:*   * *Scrape the data from the given website* * *Search within data with data filters.* * *Sort the data according to his own sorting method* * *User will be given choice of different algorithms* * *User will interact the application with proper graphical user interface (Designed with pyQT5)*   *Our system is capable of scrap data from some related website and not only scrapping but to make scrapping efficient we add a feature to pause and resume and cancel the scrapping and the progress of scrapping is shown in progress bar.*  *We allow different type of sorting like ascending and descending and the user also select the sorting algorithm of his own choice (insertion sort, bubble sort, merge sort) etc the option is given to change the algorithm.*  *We also provide different types of searching like from title of plot price area or the exact location and also with the name of dealer of the desired plot. There will a manager of our software who can use it with a user name and password we also add more manager.* |
| ***Business Case*** |  |
| Outline the business need for the project | *As today there is a big need of property everyone needs his own house and the state advisors need to take as many as they can record of properties in order to earn more so they are un able to keep recode of every new plot that is available to sell so this system will help them as it scrap latest data of plots and the not need to physically move and search plots so they just our system and by sitting in front of their computer they can see more than one million properties and also compete in market.* |
| End user of the product | * *Estate advisors* * *Person who wants to buy property* |
| Motivation for Project | *Our motivation is to learn data scrapping how we scrap information from any website and also an interesting problem of properties how can it solve problem of a lot of estate advisors.* |
| State the level of impact expected should the project proceed and implications of not proceeding | *There will be a huge impact and it will be for both the normal user and, but it will be a great benefit for the real estate advisor, he will be able to know more about the properties in his area and it can cause a great profit in his revenue. On the other hand, the normal user can view in which areas the property is god to buy.* |
| ***Technical Details*** |  |
| Name of Entity | * *Title* * *Description* * *Location* * *Area* * *Price* * *Phone Number* * *Agent Name* |
| Attributes of Entity  (Minimum seven attributes/rows can be increased) | |  |  |  | | --- | --- | --- | | **Name** | **Data Type** | **Description** | | *Title* | *String* | *The title of the property will show that with which title the person has named his property.* | | *Description* | *String* | *A brief description about the property like some additional information the buyer must know is provided in here.* | | *Location* | *String* | *The location will indicate the location of the property, where it is located across the globe.* | | *Area* | *String* | *The Area of the property will represent that how much area (It could be in any form like Marla, Kamal) the property has covered.* | | *Price* | *integer* | *Price of the property is displayed* | | *Phone#* | *integer* | *Phone number of the seller will be available* | | *Agent name* | *String* | *The name of the agent/seller will be given here.* | |
| Sample of Scrapping Source |  |
| GitHub Repository Link | *https://github.com/alitariq12369/CS261F21PID31* |
| Sorting Algorithms | * *Insertion sort* * *merge sort* * *selection sort* * *bubble sort* * *quick sort* * *radix sort* * *bucket sort* * *tree sort* * *shell sort* * *heap sort* * *radix sort* * *hybrid sort*   *A brief description on all the algorithms is given below.* |
| |  |  | | --- | --- | | **Algorithm Name** | **Description (Each algorithm in 2-3 lines)** | | *Insertion Sort* | *Insertion sort is an algorithm that sorts an array on the comparison basis. Its comparison element to its one level behind and check that if it is greater or smaller than the number. After doing all this, it inserts the element to its original position in the array. Insertion sort is easy to understand.* | | *Merge Sort* | *Merge sort is an algorithm in which we are given an unsorted array and we have to sort it by divide and conquer method using recursive technique. In this case, an array is split into two sub array and they are given the names left and right array recursively, then the base is reached where we have only one element which is sorted in it. Like this the recursion tree is formed and the complete array is sorted in this way.* | | *Selection Sort* | *This is the most interesting algorithm which I have found so far. In this algorithm we find the minimum element of the array and place it at the beginning of the array. Like, in the first iteration, the minimum is placed at first index and like this it iterates over the complete array and at the end, the complete array is sorted and this type of sorting is called selection sort.* | | *Bubble Sort* | *Bubble sort algorithm is also a comparison-based algorithm. In this algorithm, if we are given an array of n elements where n could be any positive integer. In this algorithm, each element is compared with its next element and if it is greater, than it is swapped with that element.* | | *Quick Sort* | *Quick sort algorithm is also based on divide and conquer methos which is solved recursively using the recursion tree. In this algorithm the partition of the array is performed. We select a pivot pointing any random order and partition it based on that pivot point. The one important use of it is that the pivot element is reached at its original position, which helps us. In this way the quick sort is performed.* | | *Hybrid Sort* | *Hybrid sort is an algorithm in which hybrid sort is performed using two algorithms in which the data is sorted in two ways, first half is sorted using one algorithm and the other half is sorted using the second method.* | | *Radix Sort* | *Radix sort is a sorting algorithm in which the array is sorted digit by digit starting from least significant position and moving towards most significant position. Counting sort is also used in radix sort.* | | *Heap Sort* | *Heap sort is a comparison-based sorting technique based on Binary Heap data structure. It is similar to selection sort where we first find the minimum element and place the minimum element at the beginning. We repeat the same process for the remaining elements.* | | *Bucket Sort* | *Bucket sort also called bin sort, is a*[*sorting algorithm*](https://en.wikipedia.org/wiki/Sorting_algorithm)*that works by distributing the elements of an*[*array*](https://en.wikipedia.org/wiki/Array_data_structure)*into a number of*[*buckets*](https://en.wikipedia.org/wiki/Bucket_(computing))*. Each bucket is then sorted individually, either using a different sorting algorithm, or by recursively applying the bucket sorting algorithm.* | | *Tree Sort* | *In the algorithm of tree sort, we first create a binary search tree using the elements of the array till the number of inputs and then we traverse using in-order traversal and store the elements in that sequence in the array. After performing all the above-mentioned steps, we Return the sorted array.* | | *Shell Sort* | *Shell sort is an algorithm which is very similar to insertion sort, the element of an array is moved one position ahead hen an element has to be moved far ahead, many movements are involved. The idea of shell Sort is to allow exchange of far items. 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 sub lists of every h’th element is sorted.* | | |
| Searching Algorithms | *The feature in our system requires searching of elements from a data set, So, some algorithms for searching are also mentioned below* |
| Linear Search | *In linear search, as the name indicates we search linearly in an ordered manner, one by one, we have one value which is to be searched in the data we take it and compare it one by one with all the elements and in this way, we reach the desired element.* |
| Binary Search | *Binary search involves divide and conquer method using the technique of the recursive functionality. It is good algorithm if we have already a sorted array and we go divide it in two parts recursively and then see where we have to move either left or right, in this way our work becomes easier.* |
| Jump Search | *Jump search is somehow similar to binary search in concept like we don’t search linearly, instead of this we only search by jumping to some certain values and find out the result.* |
| Searching Filters for each data type | * *String:*   1. *Search by Title*      1. *Contains*      2. *Starts with*      3. *Ends with*      4. *Equals to*   2. *Search by Location*      1. *Contains*      2. *Starts with*      3. *ends with*      4. *Equals to*   3. *Search by name of dealer*      1. *Contains*      2. *Starts with*      3. *Ends with*      4. *Equals to*   4. *Search by area*      1. *Contains*      2. *Starts with*      3. *Ends with*      4. *Equals to* * *Int:*   1. *Contains*   2. *Equals to* |
| Multi-Level Sorting | By sorting with the help of two columns like on the basis of the title and then on the price, we can multi sort the data in a given manner. |
| ***Interfaces for your project*** |  |
| *Capture2.PNG*   |  |  |  | | --- | --- | --- | | **UI Component Name** | **Type of UI component** | **Purpose of UI Component/Other details** | | Record | Tab pane | To open record screen | | algorithms | Combo box | To select a specific sorting algorithm | | Record table | Table | To store all the 7 attributes of system | | Sort | Button | Start sorting of data | | Progress | Progress bar | Show how percent data is scraped from website | | Scrap | Button | Start scrapping from website | | Pause | Button | Pause scrapping process | | Resume | Button | Resume scrapping process | | Stop | Button | Stop scrapping process | | By title | Tab pane | To open screen to search by title of plot | | By location | Tab pane | To open screen to search by location of plot | | By area | Tab pane | To open screen to search by area of plot | | By price | Tab pane | To open screen to search by price of plot | | By agent | Tab pane | To open screen to search by agent name of plot | | Enter | Text field | To input the search statement | | Search | button | Search statement in table | | Exit | button | Exit from system | | |