International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 08 Issue: 09 | Sept - 2024 SJIF Rating: 8.448 ISSN: 2582-3930

TASKMASTER: Home Service Provider Using Linear Search Algorithm

Dr.M.Sengaliappan, Ragavi M

Head of the department , Department of Computer Applications, Nehru college of management, Coimbatore, Tamilnadu, India.
II MCA, Department of Computer Applications, Nehru college of management, Coimbatore, Tamilnadu, India.

ABSTRACT:

This paper presents the plan and execution of a home fix administrations application that uses a straight quest calculation for proficient help recovery. By tending to the difficulties looked by purchasers in finding proper administrations, this task improves client experience.

The utilization of Java for execution features the combination of central calculations in true applications, especially in the home fix industry.

1. INTRODUCTION

The rising interest in home fix administrations requires effective frameworks that permit purchasers to get to pertinent administrations rapidly. Conventional techniques can be bulky and wasteful, prompting client disappointment. This study frames the improvement of an electronic application that utilizes a direct inquiry calculation to smooth out the most common way of finding home fix administrations.

By utilizing Java's abilities, the undertaking shows the way that straightforward calculations can successfully further develop administrations openness.

2. Foundation

2.1 Home Fix Administrations Industry

The home fix administrations area plays a significant part in keeping up with private properties. Purchasers face different difficulties, including administration availability, quality affirmation, and time productivity.

An effective hunt component can fundamentally improve client fulfillment by empowering speedy admittance to important administrations.

2.2 Search Algorithm

Search Algorithm are fundamental in programming improvement for recovering information productively. This paper centers around the direct pursuit calculation, one of the least complex and most clear-looking through procedures.

3. Linear Search Algorithm

3.1 Definition

Linear Search Algorithm that checks every component in a rundown successfully until a match is found or the whole rundown has been looked.

3.2 Algorithm Steps

1. Begin from the main component of the rundown.

2. Contrast the ongoing component and the objective worth.

3. On the off chance that a match is found, return the list of the ongoing component.

4. Assuming no match is found, move to the following component.

5. Rehash stages 2-4 until either a match is found or all components have been checked.

3.3 Time Complexity

The time complexity of straight pursuit is O(n), where n is the quantity of components in the rundown.

This intends that in the most dire outcome imaginable, the calculation might have to really look at each component in the rundown.

3.4 Benefits and Limits

Advantages:

- Easy to carry out and comprehend.
- Doesn't need arranged information or extra information structures.

Limitations:

- Wasteful for huge datasets contrasted with other pursuit calculations (e.g., double hunt), which require arranged information.

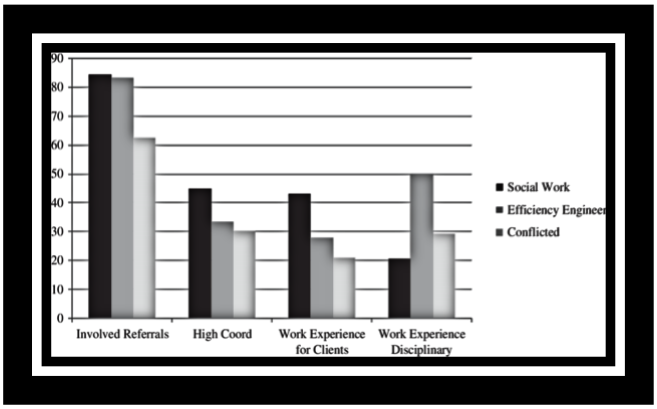
4. Application in Home Fix Administrations Framework

The home fix administrations application permits clients to look for different administrations.
(e.g., plumbing, electrical work, carpentry) utilizing a basic inquiry interface.

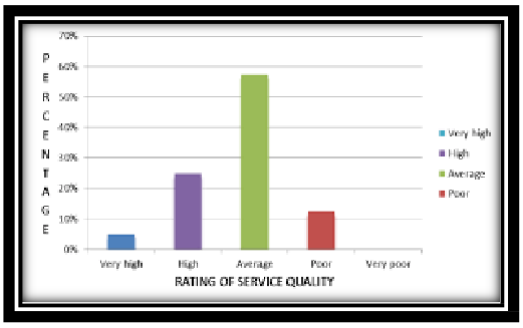
Model Situation

A client searching for "plumbing administration" can enter this term into the inquiry bar.
The application will utilize a direct inquiry calculation to repeat a rundown of accessible administration and return any matches.

BAR GRAPH FOR THE TASKER SERVICE:



BAR GRAPH FOR THE SERVICE RATING:



Code Model

```
'''java

public List<Service> linearSearch(List<Service>
administrations, String question) {

   List<Service> result = new ArrayList<>();

   for (administration : administrations) {

      on the off chance that
(service.getName().toLowerCase().contains(query.toLowerC
ase())) {

         result.add(service);

      }

   }

   bring result back;

}
```

on the off chance that (service.getName().toLowerCase().contains(query.toLowerCase()))

5.Performance Analysis

5.1 Testing Methodology

The application was tried utilizing different datasets to assess the presentation of the straight inquiry calculation.

5.2 Outcomes

Little Dataset

Execution was palatable, with reaction times well inside adequate cutoff points.

Medium Dataset: Reaction times started to increase observably, but the calculation stayed utilitarian.

Enormous Dataset:

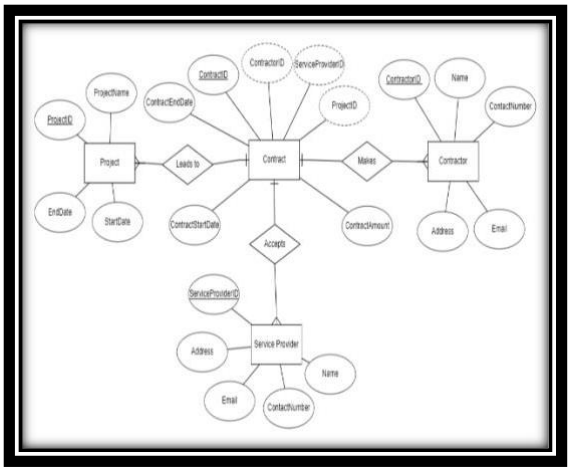
True to form, the straight hunt exhibited execution corruption, taking essentially longer to bring results back.

5.3 Conclusion from Performance Analysis

While direct pursuit is appropriate for little to medium datasets, its presentation declines as the dataset develops, demonstrating the requirement for additional productive calculations in ongoing adaptations.

6. Execution in Java

The application is fabricated utilizing Java and the Spring Boot system, which takes into account the making of Soothing administrations.



6.1 Innovation Stack

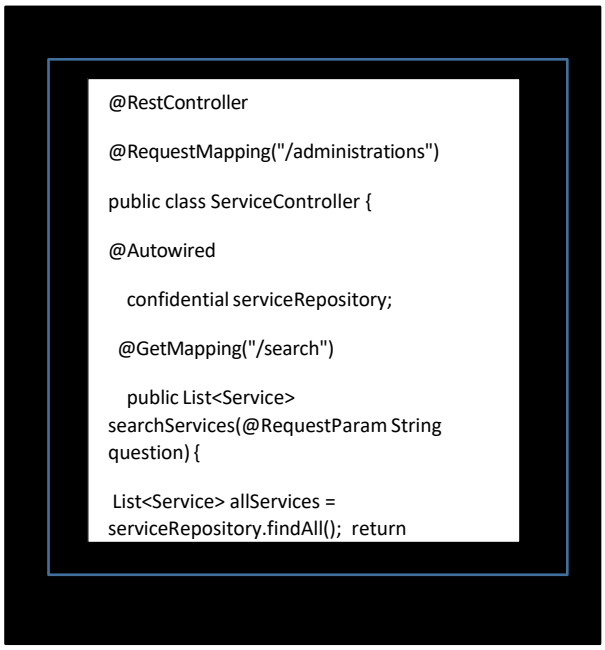
Java: For backend rationale.

Spring Boot: For making and overseeing web applications.

MySQL: For information capacity. HTML/CSS/JavaScript: For the UI.

ER DIAGRAM

6.2 Example Code Bit



```
@RestController
@RequestMapping("/administrations")
public class ServiceController {
@Autowired
   confidential serviceRepository;
 @GetMapping("/search")
   public List<Service>
searchServices(@RequestParam String
question) {
 List<Service> allServices =
serviceRepository.findAll();  return
```

7. Difficulties and Future Work

7.1 Current Difficulties

Execution Limitations: The straight inquiry calculation's failure with enormous datasets.

Scalability: As the quantity of administration expands, the application needs to successfully deal with bigger volumes of information.

7.2 Future Work

To address these difficulties, future improvements could include:

Executing More Productive Algorithms: Consider involving paired scan or listed scan strategies for bigger datasets.

Data set Optimization: Using ordering in the data set to further develop search times.

Client Experience Improvements: Upgrading the UI for better communication and availability.

8. Conclusion

The home fix administration application really shows the combination of a direct hunt calculation inside a Java structure.

While it gives a pragmatic answer for shoppers in getting to administration, its constraints with bigger datasets feature the requirement for continuous turn of events.

Future cycles ought to zero in on streamlining execution and upgrading client experience.

References

1. Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2009). *Introduction to Algorithms*. MIT Press.

2. Knuth, D. E. (1997). *The Specialty of PC Programming*. Addison-Wesley.

3. Prophet. (n.d.). Java Documentation. Recovered from https://docs.oracle.com/en/java/

4. Spring Structure Documentation. (n.d.). Recovered from https://spring.io/projects/spring-boot