DHIRUBHAI AMBANI INSTITUTE OF INFORMATION AND COMMUNICATION TECHNOLOGY

FIRST YEAR

PROJECT REPORT

ON

RESTAURANT MANAGEMENT SYSTEM

SUBMITTED BY:

SANEPARA KHUSHI SURESHBHAI : 202001152
MEHAK RAINA : 202001220

SUBMITTED TO:

PROF. ARCHANA NIGAM

ACKNOWLEDGEMENT

We would like to express our gratitude towards Prof. Archana Nigam for guiding us throughout this project and allowing us to conduct the “Restaurant Management System” project. We are thankful to our professor for providing us knowledge and understanding about object oriented programing which helped us in building our project.

PROBLEM STATEMENT:

- Create a Restaurant management system like phone application for both the owner of the restaurant as well as customers.

Admin Department:

- Owner of Restaurant can add food items in the menu as per his/her convenience and delete it if the particular food item is not available and check the menu as well as the total deals of the day.

Customer Department:

- Firstly, the customer can see the list of food items which is available and has been already decided by the admin department. Therefore, customers can order food items and check it once in the final bill provided. If they want to add or delete a particular food item, they can do the same.
- Moreover, they may receive the final bill and they can check the food items as well as their quantity and amount before paying.

APPROACH TO PROBLEM:

There are some data structures which can be used to approach this problem.

These are: Array, Linked list, Hashing Table, Stack, Queue, etc.

- **Stack**:
  Since stack follows LIFO principle, Customer and admin can delete only the items which he/she entered at the last. This is not always feasible as it may be possible that a customer or an admin may wish to delete an item randomly which will not take place in case of stack data structure.

- **Queue:**
  Since queue follows FIFO principle, Customer and admin can delete only the items which he/she entered first. This is not always feasible as it may be possible that a customer or an admin may wish to delete an item randomly which will not take place in case of queue data structure.

- **Priority Queue:**
  Customer and admin can remove items only based on priority but that will not be feasible as it may be possible that customer's priority gets changed.

- **Hashing Table:**
  In case of hashing table, we need to maintain the key and value for the customer but we came to know that maintaining the key of the customer is not necessary in this problem. But if we still do that, it will be a memory wastage to store key and will increase time complexity to do hashing and all such things.

- **Array:**
  Since array is a static data structure so we cannot increase the size of the array. Therefore, it may be possible that after execution, the customer may order beyond the size limit and admins can add items too.

- **Linked List:**
  Finally, we approached the Linked List data structure as it removes all such limitations and is much more feasible than all other data structures. We have used Singly Linked List in this program.

THINGS WE LEARNED DURING ENTIRE PROJECT:

- We learned how to think with respect to time complexity and memory complexity.

- We learned how we can make programs which is easy to understand and it can be applied in every case.

- We learned how we can abstract data from the end user. (We have made a project file for that, instead of a simple cpp file. That is use of object oriented programming.)

- We got understanding about which data structure to use when.

- Most importantly, we learned to relate applications and its' data structure.

LIMITATIONS OF OUR PROJECT:

- We have used a Linked List so it will take O(n) time to add and remove any particular food item.
- We were not able to align the food name and its' price perfectly in the case of admin add any food items.
- We could not arrange the food item's index after removing any item. (We can do it, but it will take more time).