**[Food Ordering & Delivery Management System]**

***[Ashmal Anis, 19K-0305, Section H]***

***[Hasnain Somani, 19K-0204, Section H]***

***[Abdul Samad, 19K-1396 Section H]***

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Acknowledgment**

Most of the project has been covered under the programming concepts taught in the classes. Although, to increase the attraction, and efficiency at some points in the program, few built-in/ helper functions were used from the internet. No specific website has been favored in our searches- the one with the best idea has been adopted. These websites include stackoverflow, geeksforgeeks, and tutorialspoint. The ideas taken from internet are: Regex pattern- used in verification of email, and phone number, time function to print current time, random function, and password mask function. I would also like to so thank our instructor Sir Zain who always supported us with prompt replies through email and discussions and tried to resolve our errors.

**1. Introduction**

As the name of the project briefs the function of the project, this project is in regards to food ordering, and management which is delivered to the customer. Primarily, the program prompts the user to sign up and input his/her information which includes their name, email id and password to register and login, and their address and telephone number for ease in delivery. The email id and phone number is checked using a specified regex patterns (regular expressions) to make sure the user inputs correct information. Once the user signs up, his/her information is stored in a file to make it easier for the user to login whenever they want to. The user is also given a variety of options of premium memberships available if the customer wishes to avail it.

After logging in, the customer has a list of options- the user can access our menu, and place their order. These orders can also be customized- the program outputs a list of suitable options according to the user’s order, which can be selected by the user. After placing order, and customizing it, the bill will be printed, which will be displaying the discount too incase the customer has availed the premium membership.

Apart from simply placing order, and getting it delivered, a unique approach of tracking the order has also been adopted. The program prompts the user to input their order number- which if matched, order details are outputted, as well as the rider’s name, and telephone number is provided which allows the customer to directly contact their rider. Moreover, delivery scheduling makes this project a unique idea. Delivery scheduling allows the customer to mark a date (including the current date, and future dates), and place an order to be delivered on the specified date. Customers can later login to provide their valuable feedbacks too, which are responded accordingly by the program. Customer information security has been kept as a priority, which is why a manager cannot access any customer data. They can only access the total tax and revenue generated by the company. Manager do have access to view total revenue, tax and feedbacks of the customer.

**2. Tools and technologies used**

**IDES used:**

Visual Studio Code, Dev C++(Language Standard ISO C++ 11)

**Libraries:**

* + - 1. #include <iostream>, used for cin,cout (input/output stream) to take inputs and outputs.
      2. sizeof() -> use to return size of datatypes.
      3. size() -> use to return length of string.
      4. #include <cstdlib>, used for rand() to randomly generate numbers.
      5. #include <cstdio>, >, used for fflush(stdin clear buffer from inputs.
      6. #include <fstream> , used for ifstream i("filename.txt") to read from files
      7. ofstream o("filename.txt") -> use to write in files.
      8. #include <string>/<cstring>>, used for gets() to take input of string.
      9. getline() -> use to read a string or line from input.
      10. #include <windows.h>,>, used for system("cls”) to clear screen.
      11. gotoxy(x,y) -> use to move the cursor with their respective x and y positions.
      12. #include <conio.h> used for getch(), getche() to get character and proceed
      13. #include <time.h>/#include <ctime>, >, used for 1.time\_t time() to return the current time from your device. 2.localtime() , which takes a pointer of type time\_t as its argument and returns a pointer object of structure tm.
      14. The value returned by localtime() function is the local time, then, the hours, minutes and seconds can be accessed using tm\_hour, tm\_min and tm\_sec respectively.
      15. 3.ctime() function takes a pointer to time\_t object as its parameter and returns a text representation of the form:
      16. #include <stdlib.h>, used for exit() to terminate the program
      17. calloc -> use for memory allocation
      18. #include <regex>, used for regex\_match() to match patterns with input strings.
      19. #include <exception> used to inherit exception class.

No Database but simple filing is used through txt files, or framework used.

**3. Class diagram**

**Class DeliveryServices**

-revenue: double

- tax: double

+gotoxy(int,int): void

+loading(): void

+Intro(): void

+feed\_back(): void

+check\_date(int, int ,int): static bool

+Email\_check(string): static bool

+Manager\_Portal(): static void

+mask\_password():static char\*

+check\_phone():static char\*

**Class Food**

-index: int

-category: char\*

-food\_name: char\*

-food\_name: double

**Class Customer**

-cus\_name: char\*

- email: char\*

- address: char\*

- phone: char\*

- password: char\*

- code: char\*

- validity: bool

-rider: string

-phone: string

-Time: int

**Class Delivery**

+gotoxy(int,int): void

+Schedule\_Order(Customer): Food\*

+Tracking(Food\*,int): void

+Add\_Data(): void

+Display(): void

+Show\_Food(int): void

+Food\_Customization(Food\* Final\_Array, int count): int\*

+Display2(): void

+Log\_In() : Customer

+Sign\_Up() : void

**Class MyException**

**Class Order**

-index: int

-order\_number: int

-total\_price: static double

-tax: static double

-time\_dispatch: char\*

-count: int

+what():const char\* const throw():const char\*

+gotoxy(int,int): void

+Bill(Food\*, int, int\*, Customer): void

+Order\_Now(int \*ptr, int count): Food\*

+show\_menu(Customer): Food\*

friend class DeliveryServices

* **Class DeliveryServices**: this class acts as one of the most important classes, specially for interaction with the user. This class has most of the functions that are visible to the user, such as the loading, or intro functions. Mainly, this class comprises of functions that are used multiple times, such as the check email, and phone, as well as the mask password function.
* **Class Delivery**: as stated by its name, its functionality is just the same. This class totally works on the backend working on delivery only. Its functions allow you to track your order, as well as schedule any future orders.
* **Class Customer**: this class is the backbone of the program, as it stores all of the information related to the user. The functions (sign up, and login) then allow the user to enter the required information, and process it accordingly.
* **Class Food**: this class is used to store all information, and do all working related to the menu. Food items are stored, displayed, and customized in this class.
* **Class Order**: this class is the most important class of this program. In this class, the order of a customer is stored in an array, which is returned from the function Order\_Now. Also, the total Bill is calculated, stored, and displayed in this same class. Moreover, this class has private static attributes which store the total revenue, and tax earned by the company. In short, this class has all sensitive business related information, and it is the most important part of the project.
* **Class Exception**: This class is used to inherit builtin exception class for overriding the function of what (). This class allows us to have customized what functionality.

**4. Link to source**

**GitHub Link:** <https://github.com/ashmalvayani/OOP-Project>

**Google Drive:** https://drive.google.com/file/d/1pgPC9Aw2cmEBWGgHL1dfRlzD3wALoj0I/view?usp=sharing

**5. Future work**

Firstly, including a touch of GUI would make the program more attractive, and user friendly. Including GUI in future covers another aim that is to include maps to show the location of the user, and the delivery boy. Apart from that, a 24/7 support could be added to the main menu if we knew how to efficiently respond to customer queries, and code it accordingly.

Also, filing has been useful, but database would have been a much better approach to store customer information, and the customer can get all of his orders history as soon as he logs in. Improving the Manager’s authority might also be considered in future, where we can allow the manager to access all food items, and add, update or delete any item.