**PROJECT REPORT**

**On**

**Secure Food Delivery Management System**

**Project id- 8**

**Submitted by**

Harshpreet Singh Sandhu (Enroll No. R134218065)

Neelesh Kumar (Enroll. No. R134218102)

Naman Sharma (Enroll. No. R134218099)

Karan Bhandari (Enroll. No. R134218076)

**Under the guidance of**

Mr. Keshav Kaushik

Department of Systemics



**SCHOOL OF COMPUTER SCIENCE**

UNIVERSITY OF PETROLEUM & ENERGY STUDIES

Bidholi Campus, Energy Acres, Dehradun – 248007.

**July Dec-2020**



**CANDIDATE’S DECLARATION**

We hereby certify that the project work entitled **“Secure *food delivery and management system*”**in partial fulfillment of the requirements for the award ofthe Degree of BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE AND ENGINEERING with specialization in **CYBER SECURITY AND FORENSICS** and submitted to the School of Computer Science, Department of Systemics, University of Petroleum & Energy Studies, Dehradun, is an authentic record of my/ our work carried out during a period from **August**-**2020** to **October-2020** under the supervision of **Mr. Keshav Kaushik , Dept. of Systemics.**

The matter presented in this project has not been submitted by us for the award of any other degree of this or any other University.

**Karan Bhandari** R134218076

|  |  |
| --- | --- |
| This is to certify that the above statement made by the | candidate is correct to the best of |
| my knowledge. |  |
| Date: \_\_\_\_\_\_\_\_\_\_\_\_\_2020 | **Mr. Keshav Kaushik** |
|  | Project Guide |
| **Dr. Neelu Jyoti Ahuja** |  |
| HOD – Systemics |  |
| School of Computer Science |  |
| University of Petroleum & Energy Studies |  |
| Dehradun – 248 001 (Uttarakhand) |  |
|  |  |

**Harshpreet Singh Sandhu** R134218065

**Naman. Sharma** R134218099

**Neelesh Kumar**

R134218102

**ACKNOWLEDGEMENT**

We wish to express our deep gratitude to our guide **Mr. Keshav Kaushik**, for all advice, encouragement and constant support he has given us throughout our project work. This work would not have been possible without his support and valuable suggestions.

We sincerely thank to our respected Program Head of the Department, **Dr. Manoj Kumar**, for her great support in doing our project in **Dept. of Systemics** at **SoCS**.

We are also grateful to **Dr. Manish Prateek,** Dean- SoCS, UPES for giving us the necessary facilities to carry out our project work successfully.

We would like to thank all our **friends** for their help and constructive criticism during our project work. Finally we have no words to express our sincere gratitude to our **parents** who have shown us this world and for every support they have given us

**Name:** Karan Bhandari Neelesh Kumar Harshpreet Sandhu Naman Sharma

**Roll No:** R134218076 R134218102 R134218065 R134218099

 **School of Computer Science**

**University of Petroleum & Energy Studies, Dehradun**

**PROJECT TITLE:**  Secure Food Delivery Management System

**ABSTRACT**

The project aims to bring security in the field of food delivery by creating a mechanism that is capable of encrypting & decrypting customer details and provide shortest path for the delivery. The input take details from the customer for his/her general details which in turn will be encrypted using our modified algorithm. Customer detail will be saved in a file using file handling in the encrypted form only. After successfully creating account, customer can login using the username. At login step, password will be verified with the encrypted password present in the file. Then menu will be displayed for the choice of food. Menu will provide different kind of food options to choose from. At any point of time customer can come back from the menu or continue to explore the menu. Based on the choice of food, specific time will be taken for the preparation. For delivery of food to the customer’s destination, shortest path algorithm will be implemented to deliver it in minimum time possible. At any point of time customer can see the status of the delivery guy and distance left for destination.

**KEYWORDS:** Modified algorithm for Encryption and Decryption, Implementation of Shortest path algorithm, Status of delivery at any point of time.

**INTRODUCTION**

The “Secure Food Delivery Management System” has been developed to overcome the previous problems in the manual system. This mechanism is supported to eliminate and in some cases reduce the hardships faced by the existing system. However, this programme is designed for the important need of organizations to carry out operations in a smooth and efficient manner.

The work is reduced as much as possible in order to avoid errors while entering the data. It also provides simple and various options for input of data. Nor formal language is needed for the user to use the system. Hence, proving user-friendly. Food Delivery System, as described above, can lead to error free, secure, reliable and fast management system. It can assist the user to concentrate on other activities rather than to concentrate on the record keeping. Thus it will help organizations in better utilization of resources.

Every organization, either big or small, faces challenges to overcome and managing the information. Every Food Delivery System has different Food needs, therefore we design exclusive employee management systems that are adapted to your managerial requirements. This is made to help in strategic planning, and will help you ensure that your company is equipped with exact level of information and details for your future goals. These process will ultimately allow you to better organise resources.

Every day we come across several news of Data Breach of customer data. Even the food delivery giants like Swiggy and Zomato have failed to keep the data of the customers safe.

According to the report by “Medianama”- Amazon Web Services is a common provider that many businesses like Swiggy use to store their data in such buckets for ready retrieval. This is quite useful, for example, when Swiggy wants to show users pictures of dishes in a restaurant, but doesn’t have the important server infrastructure in-house to manage tons of such photos requests every day. Since data buckets enable you link to every public element (like a food photo from a restaurant) with a standard photo URL, they are very useful when a web service is scaling.

For this scenario, Swiggy should have saved important data behind password-protected servers that should have ideally been restricted to the company. It lead them to end up putting those files on buckets that are used for public resources, which mean that anyone searching hard enough could find those files. And they did.

According to a report by “Economic Times”- Zomato has suffered a security incident with over 17 million user records stolen from the Giant Company’s database. The leaked information has email addresses and passwords that are hashed.

In regard to these breaches, we have developed our modified Encryption algorithm which will save encrypted files on our system only in order to avoid Data Breaches through servers.

**PROBLEM STATEMENT**

Creating a secure food delivery system in order to protect Customers’ privacy & minimizing the delivery time using our modified algorithm for encryption-decryption and shortest path algorithm.

**LITERATURE REVIEW**

|  |  |  |  |
| --- | --- | --- | --- |
| **Title** | **Link** | **Author** | **Remarks** |
| Cryptography | <https://www.pdfdrive.com/serious-cryptography-a-practical-introduction-to-modern-encryption-e183556059.html> | Jean-Philippe Aumasson | It explain various types of algorithms for encryption like RSA and their implementation in real life scenarios. |
| RSA | <https://www.geeksforgeeks.org/rsa-algorithm-cryptography/?ref=rp> | Mohit Gupta | This post defines the coding of RSA and implementation. |
| Algorithms used in now-a-days | <https://www.geeksforgeeks.org/encryption-its-algorithms-and-its-future/> | Jash Kothari | It explains about RSA, AES and DES algorithms. |
| RSA (cryptosystem) | <https://en.wikipedia.org/wiki/RSA_(cryptosystem)> | Wikipedia | The history behind RSA. |

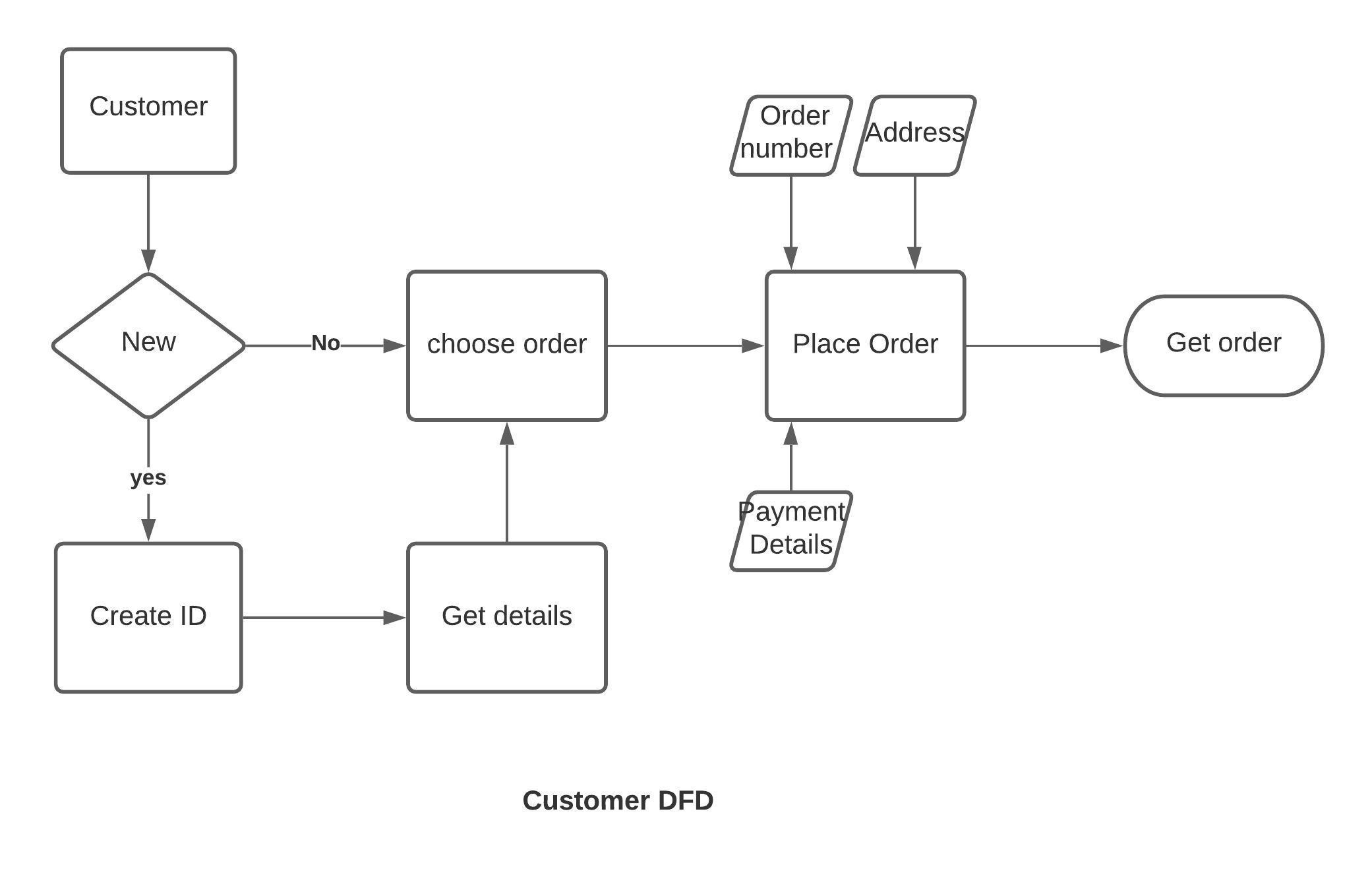
**OBJECTIVES**

To create a program for Secure Food Delivery System.

Sub Objectives

* Provide customer options for Sign-up and Login
* Get customer details and encrypt it in file
* Authenticate the customer
* Display the menu and calculate customer’s bill
* Calculate distance for delivery using shortest path and display it

**METHODOLOGY**

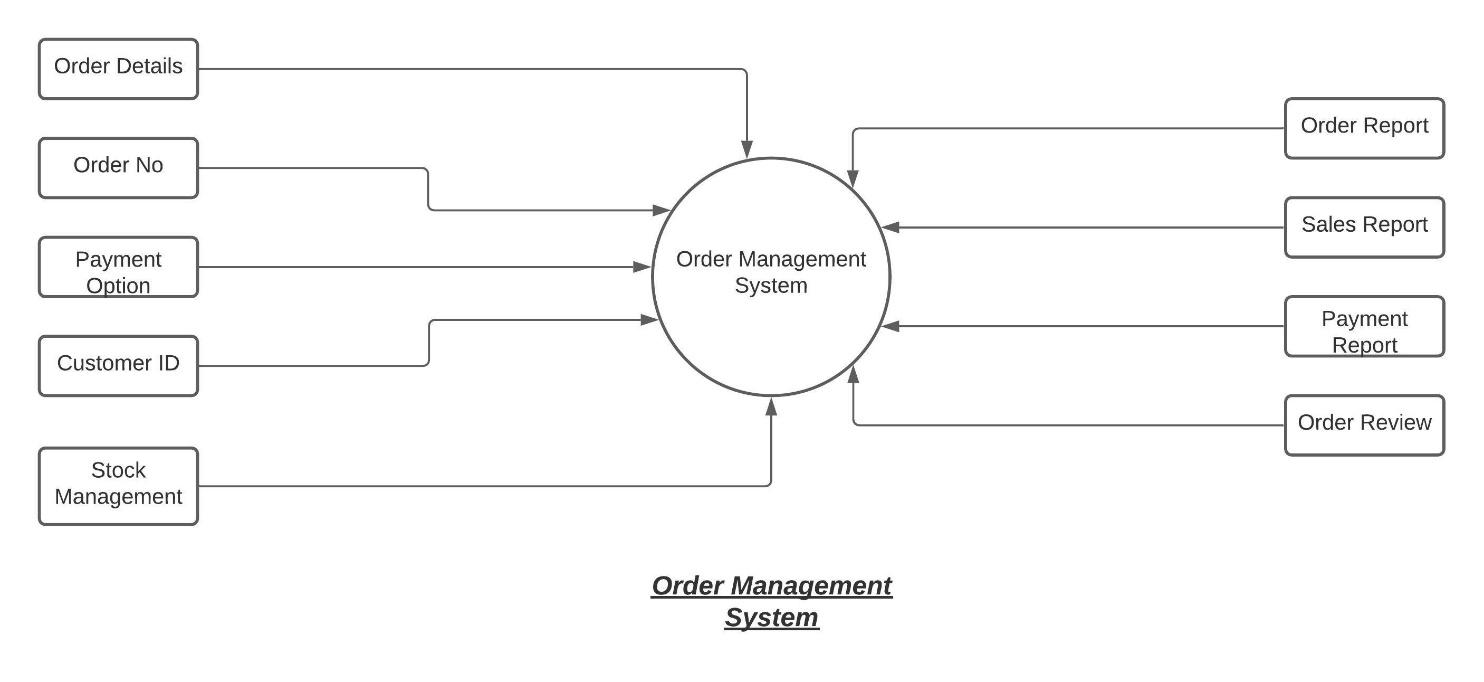


In this scenario, we have a customer either he/she can be new or existing.

1. If the customer is new, he/she have to create an ID for the food delivery system by filling up the details of him/her.
2. If the customer is already registered then no need to create the ID.

After that customer has to choose the order what he/she wants and then place the order by giving the address and payment. This will give him/her an order number.

Finally, the customer will get the order.

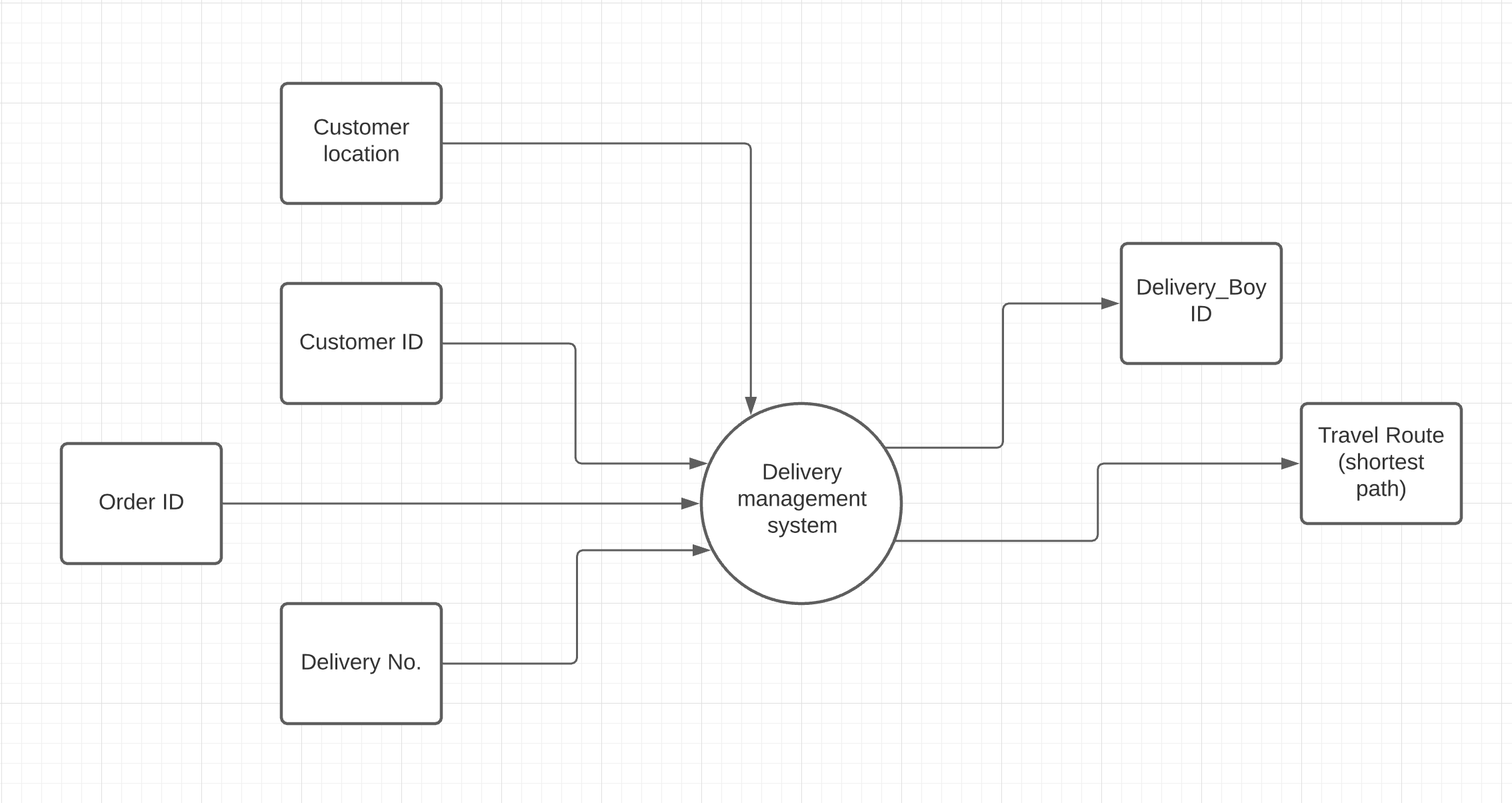


In this scenario, we are managing the order:

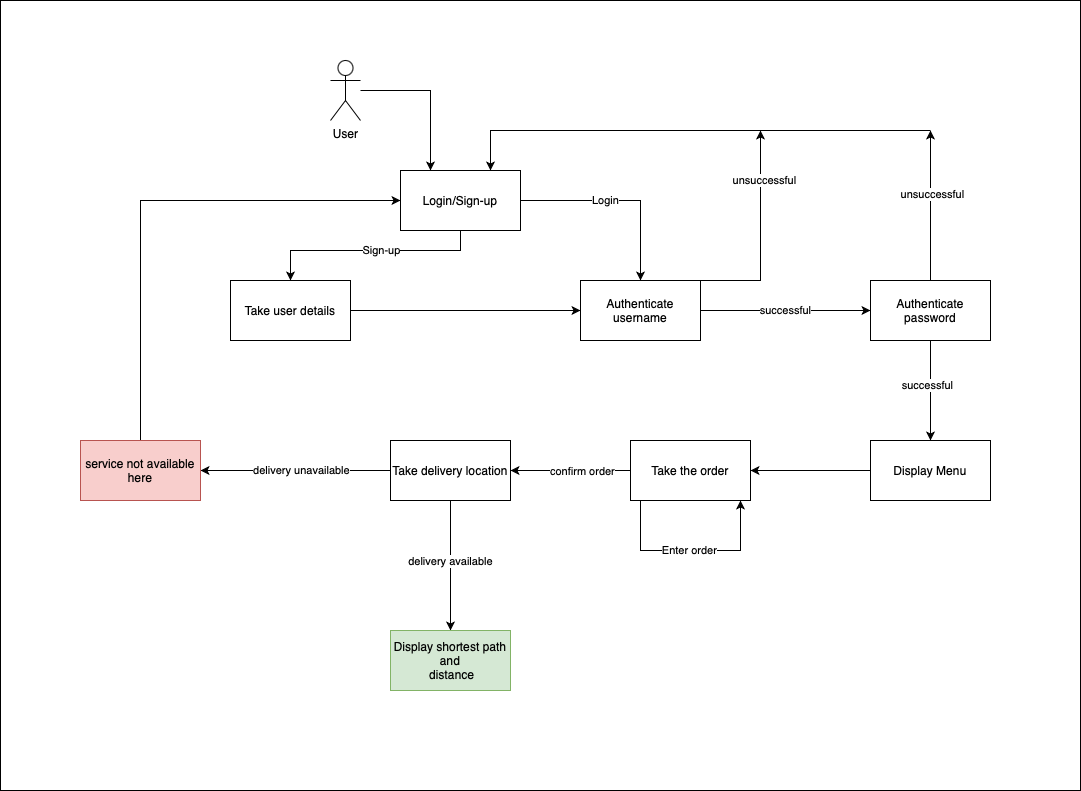
1) We get the order no then we manage the details of the order like payments, customer ID, payment option, Delivery Address and required information.

2) After doing the delivery we store the details of the delivery whether delivery is successful and store the payment information, reviews by the customer.

In payment option we check whether the customer is doing online payment or cash on delivery. Stock management is for checking whether item is available or not.

****Delivery Management System

**Workflow**



The delivery management system is more focused on our end. Here:

1. Once an order has been received, payed and the transaction is completed, an order ID is generated, which uniquely describes that particular order.
2. This order ID is further mapped with a delivery no.
3. The delivery no. is generated based on the pincode information we gather from the customer location
4. (Delivery no helps our system be more efficient as orders with similar delivery numbers can be delivered by the same delivery boy)
5. Finally, the devilry no. is assigned to a delivery\_boy ID, who is then notified with the shortest travel route calculated again with the customer location.

**SYSTEM REQUIREMENTS**

Hardware:

* RAM: 2GB
* Disk Space: 4GB

Software:

* C IDE or Dev C++

Operating System:

* Windows.

**REFERENCES**

[1] <https://www.medianama.com/2018/07/223-data-buckets-with-personal-info-from-swiggy-health-startup-and-others-exposed/>

# [2] <https://economictimes.indiatimes.com/small-biz/security-tech/security/zomato-hacked-security-breach-results-in-17-million-user-data-stolen/articleshow/58729251.cms>

# [3] <https://www.geeksforgeeks.org/>

# [4]<https://www.freeprojectz.com/premium-synopsis/synopsis-online-food-ordering-system> <http://yann.lecun.com/exdb/mnist/>

# [5] <https://en.wikipedia.org/wiki/RSA_(cryptosystem)>

[6]<https://app.lucidchart.com/documents/edit/02d37549-7141-4bfd-b633-f98a6780a72e/0_0> (DFD Creation tool)

[7] <https://www.google.com/search?q=customer+food+booking+dfd&rlz=1C1GCEA_enIN883IN883&sxsrf=ALeKk00qqOq30-c8VnHTJ3M3N7cgj0_Qdg:1598173767228&source=lnms&tbm=isch&sa=X&ved=2ahUKEwje67yd_bDrAhVS4HMBHUZHCLYQ_AUoAXoECA4QAw&biw=1536&bih=722> (DFD examples)

**Project Report Draft verified by**

**Project Guide HOD**

**(Name & Sign) (Dept. of Systemics)**

Mr. Keshav Kaushik

**Code-**

#include<stdio.h>

#include<conio.h>

#define INFINITY 9999

#define MAX 10

#include<string.h>

struct user // user login and passsword structure

{

unsigned char username[25];

unsigned char userpass[10];

};

struct username // user details structure

{

unsigned char personNamef[25];

unsigned char personNamel[25];

unsigned char Email[40];

long long phoneNo;

unsigned char address[100];

//long long cardno;

};

struct dboy{

char dusername[25];

char dpass[16];

};

struct fooditems //menu structure

{

char \*foodname[30];

int price;

int quantity;

};

FILE \*f;

void deliveryboy();

//-----------------------------------------------------------------------------------------

//Aes shuru

unsigned char s[256] =

{

0x63, 0x7C, 0x77, 0x7B, 0xF2, 0x6B, 0x6F, 0xC5, 0x30, 0x01, 0x67, 0x2B, 0xFE, 0xD7, 0xAB, 0x76,

0xCA, 0x82, 0xC9, 0x7D, 0xFA, 0x59, 0x47, 0xF0, 0xAD, 0xD4, 0xA2, 0xAF, 0x9C, 0xA4, 0x72, 0xC0,

0xB7, 0xFD, 0x93, 0x26, 0x36, 0x3F, 0xF7, 0xCC, 0x34, 0xA5, 0xE5, 0xF1, 0x71, 0xD8, 0x31, 0x15,

0x04, 0xC7, 0x23, 0xC3, 0x18, 0x96, 0x05, 0x9A, 0x07, 0x12, 0x80, 0xE2, 0xEB, 0x27, 0xB2, 0x75,

0x09, 0x83, 0x2C, 0x1A, 0x1B, 0x6E, 0x5A, 0xA0, 0x52, 0x3B, 0xD6, 0xB3, 0x29, 0xE3, 0x2F, 0x84,

0x53, 0xD1, 0x00, 0xED, 0x20, 0xFC, 0xB1, 0x5B, 0x6A, 0xCB, 0xBE, 0x39, 0x4A, 0x4C, 0x58, 0xCF,

0xD0, 0xEF, 0xAA, 0xFB, 0x43, 0x4D, 0x33, 0x85, 0x45, 0xF9, 0x02, 0x7F, 0x50, 0x3C, 0x9F, 0xA8,

0x51, 0xA3, 0x40, 0x8F, 0x92, 0x9D, 0x38, 0xF5, 0xBC, 0xB6, 0xDA, 0x21, 0x10, 0xFF, 0xF3, 0xD2,

0xCD, 0x0C, 0x13, 0xEC, 0x5F, 0x97, 0x44, 0x17, 0xC4, 0xA7, 0x7E, 0x3D, 0x64, 0x5D, 0x19, 0x73,

0x60, 0x81, 0x4F, 0xDC, 0x22, 0x2A, 0x90, 0x88, 0x46, 0xEE, 0xB8, 0x14, 0xDE, 0x5E, 0x0B, 0xDB,

0xE0, 0x32, 0x3A, 0x0A, 0x49, 0x06, 0x24, 0x5C, 0xC2, 0xD3, 0xAC, 0x62, 0x91, 0x95, 0xE4, 0x79,

0xE7, 0xC8, 0x37, 0x6D, 0x8D, 0xD5, 0x4E, 0xA9, 0x6C, 0x56, 0xF4, 0xEA, 0x65, 0x7A, 0xAE, 0x08,

0xBA, 0x78, 0x25, 0x2E, 0x1C, 0xA6, 0xB4, 0xC6, 0xE8, 0xDD, 0x74, 0x1F, 0x4B, 0xBD, 0x8B, 0x8A,

0x70, 0x3E, 0xB5, 0x66, 0x48, 0x03, 0xF6, 0x0E, 0x61, 0x35, 0x57, 0xB9, 0x86, 0xC1, 0x1D, 0x9E,

0xE1, 0xF8, 0x98, 0x11, 0x69, 0xD9, 0x8E, 0x94, 0x9B, 0x1E, 0x87, 0xE9, 0xCE, 0x55, 0x28, 0xDF,

0x8C, 0xA1, 0x89, 0x0D, 0xBF, 0xE6, 0x42, 0x68, 0x41, 0x99, 0x2D, 0x0F, 0xB0, 0x54, 0xBB, 0x16

};

unsigned char inv\_s[256] =

{

0x52, 0x09, 0x6A, 0xD5, 0x30, 0x36, 0xA5, 0x38, 0xBF, 0x40, 0xA3, 0x9E, 0x81, 0xF3, 0xD7, 0xFB,

0x7C, 0xE3, 0x39, 0x82, 0x9B, 0x2F, 0xFF, 0x87, 0x34, 0x8E, 0x43, 0x44, 0xC4, 0xDE, 0xE9, 0xCB,

0x54, 0x7B, 0x94, 0x32, 0xA6, 0xC2, 0x23, 0x3D, 0xEE, 0x4C, 0x95, 0x0B, 0x42, 0xFA, 0xC3, 0x4E,

0x08, 0x2E, 0xA1, 0x66, 0x28, 0xD9, 0x24, 0xB2, 0x76, 0x5B, 0xA2, 0x49, 0x6D, 0x8B, 0xD1, 0x25,

0x72, 0xF8, 0xF6, 0x64, 0x86, 0x68, 0x98, 0x16, 0xD4, 0xA4, 0x5C, 0xCC, 0x5D, 0x65, 0xB6, 0x92,

0x6C, 0x70, 0x48, 0x50, 0xFD, 0xED, 0xB9, 0xDA, 0x5E, 0x15, 0x46, 0x57, 0xA7, 0x8D, 0x9D, 0x84,

0x90, 0xD8, 0xAB, 0x00, 0x8C, 0xBC, 0xD3, 0x0A, 0xF7, 0xE4, 0x58, 0x05, 0xB8, 0xB3, 0x45, 0x06,

0xD0, 0x2C, 0x1E, 0x8F, 0xCA, 0x3F, 0x0F, 0x02, 0xC1, 0xAF, 0xBD, 0x03, 0x01, 0x13, 0x8A, 0x6B,

0x3A, 0x91, 0x11, 0x41, 0x4F, 0x67, 0xDC, 0xEA, 0x97, 0xF2, 0xCF, 0xCE, 0xF0, 0xB4, 0xE6, 0x73,

0x96, 0xAC, 0x74, 0x22, 0xE7, 0xAD, 0x35, 0x85, 0xE2, 0xF9, 0x37, 0xE8, 0x1C, 0x75, 0xDF, 0x6E,

0x47, 0xF1, 0x1A, 0x71, 0x1D, 0x29, 0xC5, 0x89, 0x6F, 0xB7, 0x62, 0x0E, 0xAA, 0x18, 0xBE, 0x1B,

0xFC, 0x56, 0x3E, 0x4B, 0xC6, 0xD2, 0x79, 0x20, 0x9A, 0xDB, 0xC0, 0xFE, 0x78, 0xCD, 0x5A, 0xF4,

0x1F, 0xDD, 0xA8, 0x33, 0x88, 0x07, 0xC7, 0x31, 0xB1, 0x12, 0x10, 0x59, 0x27, 0x80, 0xEC, 0x5F,

0x60, 0x51, 0x7F, 0xA9, 0x19, 0xB5, 0x4A, 0x0D, 0x2D, 0xE5, 0x7A, 0x9F, 0x93, 0xC9, 0x9C, 0xEF,

0xA0, 0xE0, 0x3B, 0x4D, 0xAE, 0x2A, 0xF5, 0xB0, 0xC8, 0xEB, 0xBB, 0x3C, 0x83, 0x53, 0x99, 0x61,

0x17, 0x2B, 0x04, 0x7E, 0xBA, 0x77, 0xD6, 0x26, 0xE1, 0x69, 0x14, 0x63, 0x55, 0x21, 0x0C, 0x7D

};

unsigned char mul2[] =

{

0x00,0x02,0x04,0x06,0x08,0x0a,0x0c,0x0e,0x10,0x12,0x14,0x16,0x18,0x1a,0x1c,0x1e,

0x20,0x22,0x24,0x26,0x28,0x2a,0x2c,0x2e,0x30,0x32,0x34,0x36,0x38,0x3a,0x3c,0x3e,

0x40,0x42,0x44,0x46,0x48,0x4a,0x4c,0x4e,0x50,0x52,0x54,0x56,0x58,0x5a,0x5c,0x5e,

0x60,0x62,0x64,0x66,0x68,0x6a,0x6c,0x6e,0x70,0x72,0x74,0x76,0x78,0x7a,0x7c,0x7e,

0x80,0x82,0x84,0x86,0x88,0x8a,0x8c,0x8e,0x90,0x92,0x94,0x96,0x98,0x9a,0x9c,0x9e,

0xa0,0xa2,0xa4,0xa6,0xa8,0xaa,0xac,0xae,0xb0,0xb2,0xb4,0xb6,0xb8,0xba,0xbc,0xbe,

0xc0,0xc2,0xc4,0xc6,0xc8,0xca,0xcc,0xce,0xd0,0xd2,0xd4,0xd6,0xd8,0xda,0xdc,0xde,

0xe0,0xe2,0xe4,0xe6,0xe8,0xea,0xec,0xee,0xf0,0xf2,0xf4,0xf6,0xf8,0xfa,0xfc,0xfe,

0x1b,0x19,0x1f,0x1d,0x13,0x11,0x17,0x15,0x0b,0x09,0x0f,0x0d,0x03,0x01,0x07,0x05,

0x3b,0x39,0x3f,0x3d,0x33,0x31,0x37,0x35,0x2b,0x29,0x2f,0x2d,0x23,0x21,0x27,0x25,

0x5b,0x59,0x5f,0x5d,0x53,0x51,0x57,0x55,0x4b,0x49,0x4f,0x4d,0x43,0x41,0x47,0x45,

0x7b,0x79,0x7f,0x7d,0x73,0x71,0x77,0x75,0x6b,0x69,0x6f,0x6d,0x63,0x61,0x67,0x65,

0x9b,0x99,0x9f,0x9d,0x93,0x91,0x97,0x95,0x8b,0x89,0x8f,0x8d,0x83,0x81,0x87,0x85,

0xbb,0xb9,0xbf,0xbd,0xb3,0xb1,0xb7,0xb5,0xab,0xa9,0xaf,0xad,0xa3,0xa1,0xa7,0xa5,

0xdb,0xd9,0xdf,0xdd,0xd3,0xd1,0xd7,0xd5,0xcb,0xc9,0xcf,0xcd,0xc3,0xc1,0xc7,0xc5,

0xfb,0xf9,0xff,0xfd,0xf3,0xf1,0xf7,0xf5,0xeb,0xe9,0xef,0xed,0xe3,0xe1,0xe7,0xe5

};

unsigned char mul\_3[] =

{

0x00,0x03,0x06,0x05,0x0c,0x0f,0x0a,0x09,0x18,0x1b,0x1e,0x1d,0x14,0x17,0x12,0x11,

0x30,0x33,0x36,0x35,0x3c,0x3f,0x3a,0x39,0x28,0x2b,0x2e,0x2d,0x24,0x27,0x22,0x21,

0x60,0x63,0x66,0x65,0x6c,0x6f,0x6a,0x69,0x78,0x7b,0x7e,0x7d,0x74,0x77,0x72,0x71,

0x50,0x53,0x56,0x55,0x5c,0x5f,0x5a,0x59,0x48,0x4b,0x4e,0x4d,0x44,0x47,0x42,0x41,

0xc0,0xc3,0xc6,0xc5,0xcc,0xcf,0xca,0xc9,0xd8,0xdb,0xde,0xdd,0xd4,0xd7,0xd2,0xd1,

0xf0,0xf3,0xf6,0xf5,0xfc,0xff,0xfa,0xf9,0xe8,0xeb,0xee,0xed,0xe4,0xe7,0xe2,0xe1,

0xa0,0xa3,0xa6,0xa5,0xac,0xaf,0xaa,0xa9,0xb8,0xbb,0xbe,0xbd,0xb4,0xb7,0xb2,0xb1,

0x90,0x93,0x96,0x95,0x9c,0x9f,0x9a,0x99,0x88,0x8b,0x8e,0x8d,0x84,0x87,0x82,0x81,

0x9b,0x98,0x9d,0x9e,0x97,0x94,0x91,0x92,0x83,0x80,0x85,0x86,0x8f,0x8c,0x89,0x8a,

0xab,0xa8,0xad,0xae,0xa7,0xa4,0xa1,0xa2,0xb3,0xb0,0xb5,0xb6,0xbf,0xbc,0xb9,0xba,

0xfb,0xf8,0xfd,0xfe,0xf7,0xf4,0xf1,0xf2,0xe3,0xe0,0xe5,0xe6,0xef,0xec,0xe9,0xea,

0xcb,0xc8,0xcd,0xce,0xc7,0xc4,0xc1,0xc2,0xd3,0xd0,0xd5,0xd6,0xdf,0xdc,0xd9,0xda,

0x5b,0x58,0x5d,0x5e,0x57,0x54,0x51,0x52,0x43,0x40,0x45,0x46,0x4f,0x4c,0x49,0x4a,

0x6b,0x68,0x6d,0x6e,0x67,0x64,0x61,0x62,0x73,0x70,0x75,0x76,0x7f,0x7c,0x79,0x7a,

0x3b,0x38,0x3d,0x3e,0x37,0x34,0x31,0x32,0x23,0x20,0x25,0x26,0x2f,0x2c,0x29,0x2a,

0x0b,0x08,0x0d,0x0e,0x07,0x04,0x01,0x02,0x13,0x10,0x15,0x16,0x1f,0x1c,0x19,0x1a

};

unsigned char mul\_9[] =

{

0x00,0x09,0x12,0x1b,0x24,0x2d,0x36,0x3f,0x48,0x41,0x5a,0x53,0x6c,0x65,0x7e,0x77,

0x90,0x99,0x82,0x8b,0xb4,0xbd,0xa6,0xaf,0xd8,0xd1,0xca,0xc3,0xfc,0xf5,0xee,0xe7,

0x3b,0x32,0x29,0x20,0x1f,0x16,0x0d,0x04,0x73,0x7a,0x61,0x68,0x57,0x5e,0x45,0x4c,

0xab,0xa2,0xb9,0xb0,0x8f,0x86,0x9d,0x94,0xe3,0xea,0xf1,0xf8,0xc7,0xce,0xd5,0xdc,

0x76,0x7f,0x64,0x6d,0x52,0x5b,0x40,0x49,0x3e,0x37,0x2c,0x25,0x1a,0x13,0x08,0x01,

0xe6,0xef,0xf4,0xfd,0xc2,0xcb,0xd0,0xd9,0xae,0xa7,0xbc,0xb5,0x8a,0x83,0x98,0x91,

0x4d,0x44,0x5f,0x56,0x69,0x60,0x7b,0x72,0x05,0x0c,0x17,0x1e,0x21,0x28,0x33,0x3a,

0xdd,0xd4,0xcf,0xc6,0xf9,0xf0,0xeb,0xe2,0x95,0x9c,0x87,0x8e,0xb1,0xb8,0xa3,0xaa,

0xec,0xe5,0xfe,0xf7,0xc8,0xc1,0xda,0xd3,0xa4,0xad,0xb6,0xbf,0x80,0x89,0x92,0x9b,

0x7c,0x75,0x6e,0x67,0x58,0x51,0x4a,0x43,0x34,0x3d,0x26,0x2f,0x10,0x19,0x02,0x0b,

0xd7,0xde,0xc5,0xcc,0xf3,0xfa,0xe1,0xe8,0x9f,0x96,0x8d,0x84,0xbb,0xb2,0xa9,0xa0,

0x47,0x4e,0x55,0x5c,0x63,0x6a,0x71,0x78,0x0f,0x06,0x1d,0x14,0x2b,0x22,0x39,0x30,

0x9a,0x93,0x88,0x81,0xbe,0xb7,0xac,0xa5,0xd2,0xdb,0xc0,0xc9,0xf6,0xff,0xe4,0xed,

0x0a,0x03,0x18,0x11,0x2e,0x27,0x3c,0x35,0x42,0x4b,0x50,0x59,0x66,0x6f,0x74,0x7d,

0xa1,0xa8,0xb3,0xba,0x85,0x8c,0x97,0x9e,0xe9,0xe0,0xfb,0xf2,0xcd,0xc4,0xdf,0xd6,

0x31,0x38,0x23,0x2a,0x15,0x1c,0x07,0x0e,0x79,0x70,0x6b,0x62,0x5d,0x54,0x4f,0x46

};

unsigned char mul\_11[] =

{

0x00,0x0b,0x16,0x1d,0x2c,0x27,0x3a,0x31,0x58,0x53,0x4e,0x45,0x74,0x7f,0x62,0x69,

0xb0,0xbb,0xa6,0xad,0x9c,0x97,0x8a,0x81,0xe8,0xe3,0xfe,0xf5,0xc4,0xcf,0xd2,0xd9,

0x7b,0x70,0x6d,0x66,0x57,0x5c,0x41,0x4a,0x23,0x28,0x35,0x3e,0x0f,0x04,0x19,0x12,

0xcb,0xc0,0xdd,0xd6,0xe7,0xec,0xf1,0xfa,0x93,0x98,0x85,0x8e,0xbf,0xb4,0xa9,0xa2,

0xf6,0xfd,0xe0,0xeb,0xda,0xd1,0xcc,0xc7,0xae,0xa5,0xb8,0xb3,0x82,0x89,0x94,0x9f,

0x46,0x4d,0x50,0x5b,0x6a,0x61,0x7c,0x77,0x1e,0x15,0x08,0x03,0x32,0x39,0x24,0x2f,

0x8d,0x86,0x9b,0x90,0xa1,0xaa,0xb7,0xbc,0xd5,0xde,0xc3,0xc8,0xf9,0xf2,0xef,0xe4,

0x3d,0x36,0x2b,0x20,0x11,0x1a,0x07,0x0c,0x65,0x6e,0x73,0x78,0x49,0x42,0x5f,0x54,

0xf7,0xfc,0xe1,0xea,0xdb,0xd0,0xcd,0xc6,0xaf,0xa4,0xb9,0xb2,0x83,0x88,0x95,0x9e,

0x47,0x4c,0x51,0x5a,0x6b,0x60,0x7d,0x76,0x1f,0x14,0x09,0x02,0x33,0x38,0x25,0x2e,

0x8c,0x87,0x9a,0x91,0xa0,0xab,0xb6,0xbd,0xd4,0xdf,0xc2,0xc9,0xf8,0xf3,0xee,0xe5,

0x3c,0x37,0x2a,0x21,0x10,0x1b,0x06,0x0d,0x64,0x6f,0x72,0x79,0x48,0x43,0x5e,0x55,

0x01,0x0a,0x17,0x1c,0x2d,0x26,0x3b,0x30,0x59,0x52,0x4f,0x44,0x75,0x7e,0x63,0x68,

0xb1,0xba,0xa7,0xac,0x9d,0x96,0x8b,0x80,0xe9,0xe2,0xff,0xf4,0xc5,0xce,0xd3,0xd8,

0x7a,0x71,0x6c,0x67,0x56,0x5d,0x40,0x4b,0x22,0x29,0x34,0x3f,0x0e,0x05,0x18,0x13,

0xca,0xc1,0xdc,0xd7,0xe6,0xed,0xf0,0xfb,0x92,0x99,0x84,0x8f,0xbe,0xb5,0xa8,0xa3

};

unsigned char mul\_13[] =

{

0x00,0x0d,0x1a,0x17,0x34,0x39,0x2e,0x23,0x68,0x65,0x72,0x7f,0x5c,0x51,0x46,0x4b,

0xd0,0xdd,0xca,0xc7,0xe4,0xe9,0xfe,0xf3,0xb8,0xb5,0xa2,0xaf,0x8c,0x81,0x96,0x9b,

0xbb,0xb6,0xa1,0xac,0x8f,0x82,0x95,0x98,0xd3,0xde,0xc9,0xc4,0xe7,0xea,0xfd,0xf0,

0x6b,0x66,0x71,0x7c,0x5f,0x52,0x45,0x48,0x03,0x0e,0x19,0x14,0x37,0x3a,0x2d,0x20,

0x6d,0x60,0x77,0x7a,0x59,0x54,0x43,0x4e,0x05,0x08,0x1f,0x12,0x31,0x3c,0x2b,0x26,

0xbd,0xb0,0xa7,0xaa,0x89,0x84,0x93,0x9e,0xd5,0xd8,0xcf,0xc2,0xe1,0xec,0xfb,0xf6,

0xd6,0xdb,0xcc,0xc1,0xe2,0xef,0xf8,0xf5,0xbe,0xb3,0xa4,0xa9,0x8a,0x87,0x90,0x9d,

0x06,0x0b,0x1c,0x11,0x32,0x3f,0x28,0x25,0x6e,0x63,0x74,0x79,0x5a,0x57,0x40,0x4d,

0xda,0xd7,0xc0,0xcd,0xee,0xe3,0xf4,0xf9,0xb2,0xbf,0xa8,0xa5,0x86,0x8b,0x9c,0x91,

0x0a,0x07,0x10,0x1d,0x3e,0x33,0x24,0x29,0x62,0x6f,0x78,0x75,0x56,0x5b,0x4c,0x41,

0x61,0x6c,0x7b,0x76,0x55,0x58,0x4f,0x42,0x09,0x04,0x13,0x1e,0x3d,0x30,0x27,0x2a,

0xb1,0xbc,0xab,0xa6,0x85,0x88,0x9f,0x92,0xd9,0xd4,0xc3,0xce,0xed,0xe0,0xf7,0xfa,

0xb7,0xba,0xad,0xa0,0x83,0x8e,0x99,0x94,0xdf,0xd2,0xc5,0xc8,0xeb,0xe6,0xf1,0xfc,

0x67,0x6a,0x7d,0x70,0x53,0x5e,0x49,0x44,0x0f,0x02,0x15,0x18,0x3b,0x36,0x21,0x2c,

0x0c,0x01,0x16,0x1b,0x38,0x35,0x22,0x2f,0x64,0x69,0x7e,0x73,0x50,0x5d,0x4a,0x47,

0xdc,0xd1,0xc6,0xcb,0xe8,0xe5,0xf2,0xff,0xb4,0xb9,0xae,0xa3,0x80,0x8d,0x9a,0x97

};

unsigned char mul\_14[] =

{

0x00,0x0e,0x1c,0x12,0x38,0x36,0x24,0x2a,0x70,0x7e,0x6c,0x62,0x48,0x46,0x54,0x5a,

0xe0,0xee,0xfc,0xf2,0xd8,0xd6,0xc4,0xca,0x90,0x9e,0x8c,0x82,0xa8,0xa6,0xb4,0xba,

0xdb,0xd5,0xc7,0xc9,0xe3,0xed,0xff,0xf1,0xab,0xa5,0xb7,0xb9,0x93,0x9d,0x8f,0x81,

0x3b,0x35,0x27,0x29,0x03,0x0d,0x1f,0x11,0x4b,0x45,0x57,0x59,0x73,0x7d,0x6f,0x61,

0xad,0xa3,0xb1,0xbf,0x95,0x9b,0x89,0x87,0xdd,0xd3,0xc1,0xcf,0xe5,0xeb,0xf9,0xf7,

0x4d,0x43,0x51,0x5f,0x75,0x7b,0x69,0x67,0x3d,0x33,0x21,0x2f,0x05,0x0b,0x19,0x17,

0x76,0x78,0x6a,0x64,0x4e,0x40,0x52,0x5c,0x06,0x08,0x1a,0x14,0x3e,0x30,0x22,0x2c,

0x96,0x98,0x8a,0x84,0xae,0xa0,0xb2,0xbc,0xe6,0xe8,0xfa,0xf4,0xde,0xd0,0xc2,0xcc,

0x41,0x4f,0x5d,0x53,0x79,0x77,0x65,0x6b,0x31,0x3f,0x2d,0x23,0x09,0x07,0x15,0x1b,

0xa1,0xaf,0xbd,0xb3,0x99,0x97,0x85,0x8b,0xd1,0xdf,0xcd,0xc3,0xe9,0xe7,0xf5,0xfb,

0x9a,0x94,0x86,0x88,0xa2,0xac,0xbe,0xb0,0xea,0xe4,0xf6,0xf8,0xd2,0xdc,0xce,0xc0,

0x7a,0x74,0x66,0x68,0x42,0x4c,0x5e,0x50,0x0a,0x04,0x16,0x18,0x32,0x3c,0x2e,0x20,

0xec,0xe2,0xf0,0xfe,0xd4,0xda,0xc8,0xc6,0x9c,0x92,0x80,0x8e,0xa4,0xaa,0xb8,0xb6,

0x0c,0x02,0x10,0x1e,0x34,0x3a,0x28,0x26,0x7c,0x72,0x60,0x6e,0x44,0x4a,0x58,0x56,

0x37,0x39,0x2b,0x25,0x0f,0x01,0x13,0x1d,0x47,0x49,0x5b,0x55,0x7f,0x71,0x63,0x6d,

0xd7,0xd9,0xcb,0xc5,0xef,0xe1,0xf3,0xfd,0xa7,0xa9,0xbb,0xb5,0x9f,0x91,0x83,0x8d

};

unsigned char rcon[11] =

{

0x8d, 0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80, 0x1b, 0x36,

};

unsigned char \* g (unsigned char wInput[4], int counter)

{

int i;

unsigned char \* wReady = malloc(4);

unsigned char temp[4] = "";

unsigned char a = wInput[0];

for(i =0;i<3; i++)

{

temp[i] = wInput[(i+1)];

}

temp[3] = a;

for (i =0; i<4;i++)

temp[i] = s[temp[i]];

//unsigned char array formed for xoring with rcon

unsigned char array2[4] = "";

array2[0] = rcon[counter];

array2[1] = array2[2] = array2[3] = 0x00;

for (i=0;i<4;i++)

wReady[i] = temp[i] ^ array2[i];

return wReady;

}

unsigned char \* keyExpansion(unsigned char key[16])

{

int i,j,k,l,m,n;

unsigned char words[44][4];

for (i = 0; i < 44; ++i)

{

for (j = 0; j <4; ++j)

{

words[i][j]=0x00;

}

}

unsigned char \* expandedKey = malloc(176);

int byteCount = 0; //this is to keep a count on the bytes of the expandedKey array

for (i=0;i<16;i++)

expandedKey[i] = key[i];

for(j=0;j<4;j++)

{

for(k=0;k<4;k++)

{

words[j][k] = expandedKey[byteCount];

byteCount++;

}

}

for(l=4;l<44;l++)

{

if((l%4)==0)

{

for(m=0;m<4;m++)

{

words[l][m] = words[(l-4)][m] ^ g(words[l-1], (l/4))[m];

}

}

else

{

for(n=0;n<4;n++)

{

words[l][n] = words[l-1][n] ^ words[l-4][n];

}

}

}

int loc=0;

for(i=0;i<44;i++ )

{

for(j=0;j<4;j++)

{

expandedKey[loc] = words[i][j];

loc++;

}

}

return expandedKey;

}

void mixColumns(unsigned char \* plainText)

{

int i;

unsigned char \* tempC = malloc(16);

for (i = 0; i < 4; ++i)

{

tempC[(4\*i)+0] = (unsigned char) (mul2[plainText[(4\*i)+0]] ^ mul\_3[plainText[(4\*i)+1]] ^ plainText[(4\*i)+2] ^ plainText[(4\*i)+3]);

tempC[(4\*i)+1] = (unsigned char) (plainText[(4\*i)+0] ^ mul2[plainText[(4\*i)+1]] ^ mul\_3[plainText[(4\*i)+2]] ^ plainText[(4\*i)+3]);

tempC[(4\*i)+2] = (unsigned char) (plainText[(4\*i)+0] ^ plainText[(4\*i)+1] ^ mul2[plainText[(4\*i)+2]] ^ mul\_3[plainText[(4\*i)+3]]);

tempC[(4\*i)+3] = (unsigned char) (mul\_3[plainText[(4\*i)+0]] ^ plainText[(4\*i)+1] ^ plainText[(4\*i)+2] ^ mul2[plainText[(4\*i)+3]]);

}

for (i = 0; i < 16; ++i)

{

plainText[i] = tempC[i];

}

free(tempC);

}

void inverseMixedColumn (unsigned char \* plainText)

{

int i;

unsigned char \* tempC = malloc(18);

for (i = 0; i < 4; ++i)

{

tempC[(4\*i)+0] = (unsigned char) (mul\_14[plainText[(4\*i)+0]] ^ mul\_11[plainText[(4\*i)+1]] ^ mul\_13[plainText[(4\*i)+2]] ^ mul\_9[plainText[(4\*i)+3]]);

tempC[(4\*i)+1] = (unsigned char) (mul\_9[plainText[(4\*i)+0]] ^ mul\_14[plainText[(4\*i)+1]] ^ mul\_11[plainText[(4\*i)+2]] ^ mul\_13[plainText[(4\*i)+3]]);

tempC[(4\*i)+2] = (unsigned char) (mul\_13[plainText[(4\*i)+0]] ^ mul\_9[plainText[(4\*i)+1]] ^ mul\_14[plainText[(4\*i)+2]] ^ mul\_11[plainText[(4\*i)+3]]);

tempC[(4\*i)+3] = (unsigned char) (mul\_11[plainText[(4\*i)+0]] ^ mul\_13[plainText[(4\*i)+1]] ^ mul\_9[plainText[(4\*i)+2]] ^ mul\_14[plainText[(4\*i)+3]]);

}

for (i = 0; i < 16; ++i)

{

plainText[i] = tempC[i];

}

free(tempC);

}

void byteSubShiftRow(unsigned char \* state)

{

int i;

unsigned char tmp[16];

tmp[0] = s[state[0]];

tmp[1] = s[state[5]];

tmp[2] = s[state[10]];

tmp[3] = s[state[15]];

tmp[4] = s[state[4]];

tmp[5] = s[state[9]];

tmp[6] = s[state[14]];

tmp[7] = s[state[3]];

tmp[8] = s[state[8]];

tmp[9] = s[state[13]];

tmp[10] = s[state[2]];

tmp[11] = s[state[7]];

tmp[12] = s[state[12]];

tmp[13] = s[state[1]];

tmp[14] = s[state[6]];

tmp[15] = s[state[11]];

for(i=0;i<16;i++)

{

state[i] = tmp[i];

}

}

void inverseByteSubShiftRow(unsigned char \* plainText)

{

int i;

unsigned char \* temp = malloc(16);

temp[0] = inv\_s[plainText[0]];

temp[1] = inv\_s[plainText[13]];

temp[2] = inv\_s[plainText[10]];

temp[3] = inv\_s[plainText[7]];

temp[4] = inv\_s[plainText[4]];

temp[5] = inv\_s[plainText[1]];

temp[6] = inv\_s[plainText[14]];

temp[7] = inv\_s[plainText[11]];

temp[8] = inv\_s[plainText[8]];

temp[9] = inv\_s[plainText[5]];

temp[10] = inv\_s[plainText[2]];

temp[11] = inv\_s[plainText[15]];

temp[12] = inv\_s[plainText[12]];

temp[13] = inv\_s[plainText[9]];

temp[14] = inv\_s[plainText[6]];

temp[15] = inv\_s[plainText[3]];

for (i = 0; i < 16; ++i)

plainText[i] = temp[i];

free(temp);

}

void AESEncryption(unsigned char \* plainText, unsigned char \* expandedKey, unsigned char \* cipher)

{

int i,rounds;

unsigned char \* state = malloc(16);

//unsigned char \* expandedKey = malloc(176);

//expandedKey = keyExpansion(Key);

//key addition for the first round

for (i = 0; i < 16; ++i)

{

state[i] = plainText[i] ^ expandedKey[i];

}

//now the 9 rounds begin

for(rounds = 1; rounds<10; rounds++)

{

byteSubShiftRow(state);

mixColumns(state);

int counter = 0;

int loc = rounds\*16;

while(counter<16)

{

state[counter] ^= expandedKey[loc];

loc++;

counter++;

}

}

//10th round

byteSubShiftRow(state);

for(i=0; i<16;i++)

{

cipher[i] = state[i] ^ expandedKey[i+160];

}

free(state);

}

void AESDecryption(unsigned char \* cipher, unsigned char \* expandedKey, unsigned char \* plainText)

{

int i,rounds;

unsigned char \* state = malloc(16);

//key whitening

for (i = 0; i < 16; ++i)

state[i] = cipher[i] ^ expandedKey[160+i];

// 9 rounds of decryption

for (rounds = 9; rounds >0 ; rounds--)

{

inverseByteSubShiftRow(state);

int counter = 0;

int loc = 16\*rounds;

while(counter<16)

{

state[counter] ^= expandedKey[loc];

loc++;

counter++;

}

inverseMixedColumn(state);

}

//final 10th round of decryption

inverseByteSubShiftRow(state);

for(i =0; i<16; i++)

plainText[i] = state[i] ^ expandedKey[i];

free(state);

}

//AES khatam--------------------------------------------------------------------------------------------------------------------

//Dijkastra Algorithm Starts---------------------------------------------------------------------------------------------------

void dijkstra(int Graph[MAX][MAX],int n,int startnode)

{

int cost[MAX][MAX],distance[MAX],pred[MAX];

int visited[MAX],count,mindistance,nextnode,i,j;

for(i=0;i<n;i++)

for(j=0;j<n;j++)

if(Graph[i][j]==0)

cost[i][j]=INFINITY;

else

cost[i][j]=Graph[i][j];

for(i=0;i<n;i++)

{

distance[i]=cost[startnode][i];

pred[i]=startnode;

visited[i]=0;

}

distance[startnode]=0;

visited[startnode]=1;

count=1;

while(count<n-1)

{

mindistance=INFINITY;

for(i=0;i<n;i++)

if(distance[i]<mindistance&&!visited[i])

{

mindistance=distance[i];

nextnode=i;

}

visited[nextnode]=1;

for(i=0;i<n;i++)

if(!visited[i])

if(mindistance+cost[nextnode][i]<distance[i])

{

distance[i]=mindistance+cost[nextnode][i];

pred[i]=nextnode;

}

count++;

}

// for(i=0;i<n;i++)

i=n-1;

// printf("%d\n",i);

if(i!=startnode)

{

if(i==1)

{

printf("\nShortest Distance to Prem Nagar from Bidholi: %d\n\n",distance[i]);

// printf("\nPath=%d",i);

j=i;

printf("Prem Nagar");

do

{

j=pred[j];

// printf("%d\n",j);

if(j==0)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-Bidholi");

}

else if(j==1)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-Premnagar");

}

else if(j==2)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-Railway Station");

}

else if(j==3)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-ISBT");

}

else if(j==4)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-Rajpur Road");

}

else if(j==5)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-Ballupur");

}

}while(j!=startnode);

}

if(i==2)

{

printf("\nShortest Distance to Railway Station from Bidholi: %d\n\n",distance[i]);

// printf("\nPath=%d",i);

j=i;

printf("Railway Station");

do

{

j=pred[j];

// printf("%d\n",j);

if(j==0)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-Bidholi");

}

else if(j==1)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-Premnagar");

}

else if(j==2)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-Railway Station");

}

else if(j==3)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-ISBT");

}

else if(j==4)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-Rajpur Road");

}

else if(j==5)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-Ballupur");

}

}while(j!=startnode);

}

if(i==3)

{

printf("\nShortest Distance to ISBT from Bidholi: %d\n\n",distance[i]);

// printf("\nPath=%d",i);

j=i;

printf("ISBT");

do

{

j=pred[j];

// printf("%d\n",j);

if(j==0)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-Bidholi");

}

else if(j==1)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-Premnagar");

}

else if(j==2)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-Railway Station");

}

else if(j==3)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-ISBT");

}

else if(j==4)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-Rajpur Road");

}

else if(j==5)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-Ballupur");

}

}while(j!=startnode);

}

if(i==4)

{

printf("\nShortest Distance to Rajpur Road from Bidholi: %d\n\n",distance[i]);

// printf("\nPath=%d",i);

j=i;

printf("Rajpur Road");

do

{

j=pred[j];

// printf("%d\n",j);

if(j==0)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-Bidholi");

}

else if(j==1)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-Premnagar");

}

else if(j==2)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-Railway Station");

}

else if(j==3)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-ISBT");

}

else if(j==4)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-Rajpur Road");

}

else if(j==5)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-Ballupur");

}

}while(j!=startnode);

}

if(i==5)

{

printf("\nShortest Distance to Ballupur from Bidholi: %d\n\n",distance[i]);

// printf("\nPath=%d",i);

j=i;

printf("Ballupur");

do

{

j=pred[j];

// printf("%d\n",j);

if(j==0)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-Bidholi");

}

else if(j==1)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-Premnagar");

}

else if(j==2)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-Railway Station");

}

else if(j==3)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-ISBT");

}

else if(j==4)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-Rajpur Road");

}

else if(j==5)

{

printf("<-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("-");

sleep(1);

printf("<-Ballupur");

}

}while(j!=startnode);

}

printf("\n\nThank You For visiting\n");

printf("\nVisit Us Again\n");

}

}

void map(char location[]) //Displaying Map

{

printf(" (4) (5) \n");

printf(" \*[RajpurRD] \* \* \* 7 \* \* \* [Ballupur] \n");

printf(" \* \* \n");

printf(" \* \* \n");

printf(" 20 \* \n");

printf(" \* \* \n");

printf(" \* \* \n");

printf(" \* \* \n");

printf(" [Bidholi] \* \n");

printf(" (0) \* \* (3) \n");

printf(" \* \* [ISBT] \n");

printf(" \* \* \* \* \n");

printf(" \* 11 \* \* \n");

printf(" \* \* \* \* \n");

printf(" \* \* \* \* \n");

printf(" 8 \* 6 \* \n");

printf(" \* \* \* \* \n");

printf(" \* \* \* 3 \n");

printf(" \* \* \* \* \n");

printf(" \* \* \* \* \n");

printf(" \* \* \* \* \n");

printf(" [Premnagar] \* \* \* \* \* 4 \* \* \* \* \* [RailwayStation] \n");

printf(" (1) (2) \n");

printf("\n\n\n\n");

int Graph[MAX][MAX],i,j,n=6,u=0;

Graph[0][0]=0;

Graph[0][1]=8;

Graph[0][2]=0;

Graph[0][3]=0;

Graph[0][4]=20;

Graph[0][5]=0;

Graph[1][0]=8;

Graph[1][1]=0;

Graph[1][2]=4;

Graph[1][3]=6;

Graph[1][4]=11;

Graph[1][5]=0;

Graph[2][0]=0;

Graph[2][1]=4;

Graph[2][2]=0;

Graph[2][3]=3;

Graph[2][4]=0;

Graph[2][5]=0;

Graph[3][0]=0;

Graph[3][1]=6;

Graph[3][2]=3;

Graph[3][3]=0;

Graph[3][4]=0;

Graph[3][5]=0;

Graph[4][0]=20;

Graph[4][1]=11;

Graph[4][2]=0;

Graph[4][3]=0;

Graph[4][4]=0;

Graph[4][5]=7;

Graph[5][0]=0;

Graph[5][1]=0;

Graph[5][2]=0;

Graph[5][3]=0;

Graph[5][4]=7;

Graph[5][5]=0;

printf("Location choosed: %s\n",location);

printf("\n");

if(strcmp(location,"Premnagar")==0)

dijkstra(Graph,2,0);

else if(strcmp(location,"RailwayStation")==0)

dijkstra(Graph,3,0);

else if(strcmp(location,"ISBT")==0)

dijkstra(Graph,4,0);

else if(strcmp(location,"Rajpurroad")==0)

dijkstra(Graph,5,0);

else if(strcmp(location,"Ballupur")==0)

dijkstra(Graph,6,0);

else

{

printf("Service not available here\n");

}

}

//Dijkastra Algorithm End------------------------------------------------------------------------------------------------------

void menu() //Menu card display

{

// struct fooditems foodlist[100];

int ch;

int totalcost=0;

//string snacks[100]={"samosa","Noodles","Nachos","Burger","Sandwich","pizza","Chicken Roll"};

// int snacksprice[100]={10,99,30,70,40,150,60};

// struct fooditems food\_menu[21];

// char main\_menu[100]={"samosa","Noodles","Nachos","Burger","Sandwich","pizza","Chicken Roll","Rajma Chawal","Matar Paneer","khadai Paneer","Daal Makhni","Roti","Veg Pulaw","Egg curry","Chicken tandoor","Butter Chicken Masala","Pastry","Choclate Browine","Cake","Milk Shake","Ice Cream"};

int main\_menuprice[21]={10,99,30,70,40,150,60,125,90,110,80,10,65,70,130,160,50,75,250,60,60};

int totalitem=0;

FILE \*f;

f=fopen("order\_review.txt","r");

int orderno=10000;

int size;

while(1)

{

if(feof(f))

break;

int d;

fscanf(f,"%lld",&d);

// printf("%lld\n",d);

orderno=d;

// printf("%lld\n",orderno);

fscanf(f, "%\*[^\n]\n");

}

fclose(f);

// printf("%d",orderno);

f=fopen("order\_review.txt","a+");

printf(" Main Menu\n");

int keynum=1;

int total\_amount=0;

int flag=0;

while(keynum<=21 && keynum!=0)

{

printf("Item Cost\n");

int i;

int quantity;

for(i=0;i<30;i++)

printf("~-");

printf("\n");

printf("\n1)Samosa - 10\n2)Noodles - 99\n3)Nachos - 30\n4)Burger - 70\n5)Sandwich - 40\n6)Pizza - 150\n7)Chicken Roll - 60\n8)Rajma Chawal - 125\n9)Matar Paneer - 90\n10)Khadai Paneer - 110\n11)Daal Makhni - 80\n12)Roti - 10\n13)Veg Pulao - 65\n14)Egg Curry - 70\n15)Chicken Tandoor - 130\n16)Butter Chicken Masala - 160\n17)Pastry - 50\n18)Chocolate Brownie - 75\n19)Cake - 250\n20)Milk Shake - 60\n21)Ice Cream - 60\n22)Exit - 0");

printf("\n\nChoice: ");

scanf("%d",&keynum);

if(keynum!=0 && keynum<=21)

{

printf("Enter quantity of item: ");

scanf("%d",&quantity);

if(flag==0)

{

orderno++;

fprintf(f,"%d ",orderno);

flag++;

}

if(keynum-1==0)

{

total\_amount += main\_menuprice[keynum-1]\*quantity;

fprintf(f,"Samosa %d %d ",main\_menuprice[keynum-1],quantity);

totalitem++;

}

else if(keynum-1==1)

{

totalitem++;

total\_amount += main\_menuprice[keynum-1]\*quantity;

fprintf(f,"Noodles %d %d ",main\_menuprice[keynum-1],quantity);

}

else if(keynum-1==2)

{

totalitem++;

total\_amount += main\_menuprice[keynum-1]\*quantity;

fprintf(f,"Nachos %d %d ",main\_menuprice[keynum-1],quantity);

}

else if(keynum-1==3)

{

totalitem++;

total\_amount += main\_menuprice[keynum-1]\*quantity;

fprintf(f,"Burger %d %d ",main\_menuprice[keynum-1],quantity);

}

else if(keynum-1==4)

{

totalitem++;

total\_amount += main\_menuprice[keynum-1]\*quantity;

fprintf(f,"Sandwich %d %d ",main\_menuprice[keynum-1],quantity);

}

else if(keynum-1==5)

{

totalitem++;

total\_amount += main\_menuprice[keynum-1]\*quantity;

fprintf(f,"Pizza %d %d ",main\_menuprice[keynum-1],quantity);

}

else if(keynum-1==6)

{

totalitem++;

total\_amount += main\_menuprice[keynum-1]\*quantity;

fprintf(f,"Chicken\_Roll %d %d ",main\_menuprice[keynum-1],quantity);

}

else if(keynum-1==7)

{

totalitem++;

total\_amount += main\_menuprice[keynum-1]\*quantity;

fprintf(f,"Rajma\_Chawal %d %d ",main\_menuprice[keynum-1],quantity);

}

else if(keynum-1==8)

{

totalitem++;

total\_amount += main\_menuprice[keynum-1]\*quantity;

fprintf(f,"Matar\_Paneer %d %d ",main\_menuprice[keynum-1],quantity);

}

else if(keynum-1==9)

{

totalitem++;

total\_amount += main\_menuprice[keynum-1]\*quantity;

fprintf(f,"Khadai\_Paneer %d %d ",main\_menuprice[keynum-1],quantity);

}

else if(keynum-1==10)

{

totalitem++;

total\_amount += main\_menuprice[keynum-1]\*quantity;

fprintf(f,"Daal\_Makhni %d %d ",main\_menuprice[keynum-1],quantity);

}

else if(keynum-1==11)

{

totalitem++;

total\_amount += main\_menuprice[keynum-1]\*quantity;

fprintf(f,"Roti %d %d ",main\_menuprice[keynum-1],quantity);

}

else if(keynum-1==12)

{

totalitem++;

total\_amount += main\_menuprice[keynum-1]\*quantity;

fprintf(f,"Veg\_Pulao %d %d ",main\_menuprice[keynum-1],quantity);

}

else if(keynum-1==13)

{

totalitem++;

total\_amount += main\_menuprice[keynum-1]\*quantity;

fprintf(f,"Egg\_Curry %d %d ",main\_menuprice[keynum-1],quantity);

}

else if(keynum-1==14)

{

totalitem++;

total\_amount += main\_menuprice[keynum-1]\*quantity;

fprintf(f,"Chicken\_Tandoor %d %d ",main\_menuprice[keynum-1],quantity);

}

else if(keynum-1==15)

{

totalitem++;

total\_amount += main\_menuprice[keynum-1]\*quantity;

fprintf(f,"Butter\_Chicken Masala %d %d ",main\_menuprice[keynum-1],quantity);

}

else if(keynum-1==16)

{

totalitem++;

total\_amount += main\_menuprice[keynum-1]\*quantity;

fprintf(f,"Pastry %d %d ",main\_menuprice[keynum-1],quantity);

}

else if(keynum-1==17)

{

totalitem++;

total\_amount += main\_menuprice[keynum-1]\*quantity;

fprintf(f,"Chocolate\_Brownie %d %d ",main\_menuprice[keynum-1],quantity);

}

else if(keynum-1==18)

{

totalitem++;

total\_amount += main\_menuprice[keynum-1]\*quantity;

fprintf(f,"Cake %d %d ",main\_menuprice[keynum-1],quantity);

}

else if(keynum-1==19)

{

totalitem++;

total\_amount += main\_menuprice[keynum-1]\*quantity;

fprintf(f,"Milk\_Shake %d %d ",main\_menuprice[keynum-1],quantity);

}

else if(keynum-1==20)

{

totalitem++;

total\_amount += main\_menuprice[keynum-1]\*quantity;

fprintf(f,"Ice\_Cream %d %d ",main\_menuprice[keynum-1],quantity);

}

}

else

{

break;

}

}

fprintf(f,"\n");

fclose(f);

int i;

for(i=0;i<30;i++)

printf("~-");

printf("\n");

f=fopen("order\_review.txt","r");

while(1)

{

int d;

fscanf(f,"%d",&d);

if(orderno==d)

{

printf("Order NO: %d\n",orderno);

printf("Item Cost Quantity\n");

while(totalitem!=0)

{

// printf("amanans");

char itemname[25];

int itemcost;

int itemquantity;

fscanf(f,"%s",&itemname);

fscanf(f,"%d",&itemcost);

fscanf(f,"%d",&itemquantity);

totalitem--;

printf("%s %d %d\n",itemname,itemcost,itemquantity);

}

break;

}

else

{

fscanf(f, "%\*[^\n]\n");

}

if(feof(f))

break;

}

printf("\nTotal Amount: %d\n",total\_amount);

for(i=0;i<30;i++)

printf("~-");

printf("\n");

deliveryboy();

}

void createAcc() //new account creation including file handling

{

int i;

struct username s;

struct user u;

//Aes ki key and values define ki h............................

unsigned char key[] = "87";

unsigned char \* expandedKey = keyExpansion(key);

unsigned char \* username = malloc(16);

unsigned char \* userpass = malloc(16);

unsigned char \* email = malloc(16);

unsigned char \* address = malloc(16);

unsigned char \* firstname = malloc(16);

unsigned char \* lastname = malloc(16);

unsigned char \* new = malloc(16);

//aes ki key khatam..................

f=fopen("userinfo.txt","a+");

printf("Username: "); //username

scanf("%s",&u.username);

FILE \*f1;

f1=fopen("decryptedfile.txt","r");

//------------------------------------------------

while(1)

{

char uname[25];

fscanf(f1,"%s",uname);

if(strcmp(uname,u.username)==0)

{

printf("\n\n ------------------------\n");

printf(" |Username Already Exist| \n");

printf(" ------------------------\n");

break;

}

else

{

fscanf(f1, "%\*[^\n]\n");

}

if(feof(f1))

break;

}

//---------------------------------------------------

// printf("%s",u.username);

if(feof(f1))

{

fclose(f1);

AESEncryption(u.username, expandedKey, username);

for (i = 0; i < 16; ++i)

{

fprintf(f,"%02X", username[i] );

}

fprintf(f," ");

printf("User password: "); //userpass

scanf("%s",&u.userpass);

AESEncryption(u.userpass, expandedKey, userpass);

for (i = 0; i < 16; ++i)

{

fprintf(f,"%02X", userpass[i] );

}

fprintf(f," ");

printf("User Phone no: "); //phoneno

scanf("%lld",&s.phoneNo);

fprintf(f,"%lld",s.phoneNo);

fprintf(f," ");

printf("User First name: "); //first name

scanf("%s",&s.personNamef);

AESEncryption(s.personNamef, expandedKey, firstname);

for (i = 0; i < 16; ++i)

{

fprintf(f,"%02X", firstname[i] );

}

fprintf(f," ");

printf("User Last name: "); //lastname

scanf("%s",&s.personNamel);

AESEncryption(s.personNamel, expandedKey, lastname);

for (i = 0; i < 16; ++i)

{

fprintf(f,"%02X", lastname[i] );

}

fprintf(f," ");

printf("User Email: "); //email

scanf("%s",&s.Email);

AESEncryption(s.Email, expandedKey, email);

for (i = 0; i < 16; ++i)

{

fprintf(f,"%02X", email[i] );

}

fprintf(f," ");

printf("User Address: "); //address

scanf("%s",s.address);

AESEncryption(s.address, expandedKey, address);

for (i = 0; i < 16; ++i)

{

fprintf(f,"%02X", address[i] );

}

fprintf(f,"\n");

}

fclose(f);

f=fopen("decryptedfile.txt","a+");

char output[100];

AESDecryption(username, expandedKey, output);

// printf("%s",output);

fprintf(f,"%s",output);

AESDecryption(userpass, expandedKey, output);

// printf("1 %s \n",usernamecheck);

// printf("%s",outPut);

// printf("%s",output);

fprintf(f," %s",output);

// char output[100];

AESDecryption(firstname, expandedKey, output);

// printf("1 %s \n",usernamecheck);

// printf("%s",outPut);

// printf("%s",output);

fprintf(f," %s",output);

// char output[100];

AESDecryption(lastname, expandedKey, output);

// printf("1 %s \n",usernamecheck);

// printf("%s",outPut);

// printf("%s",output);

fprintf(f," %s",output);

// char output[100];

fprintf(f," %lld",s.phoneNo);

AESDecryption(email, expandedKey, output);

// printf("1 %s \n",usernamecheck);

// printf("%s",outPut);

// printf("%s",output);

fprintf(f," %s",output);

// char output[100];

AESDecryption(address, expandedKey, output);

// printf("1 %s \n",usernamecheck);

// printf("%s",outPut);

// printf("%s",output);

fprintf(f," %s",output);

fprintf(f,"\n");

// fprintf(f," %s %s %s %lld %s %s \n",u.userpass,s.personNamef,s.personNamel,s.phoneNo,s.Email,s.address);

fclose(f);

}

void login() //old user login with filehandling

{

int i;

struct user s;

char uname[25];

printf("\nEnter User name: ");

scanf("%s",&uname);

f=fopen("decryptedfile.txt","r");

while(1)

{

//unsigned char usernamecheck[25];

char username[25];

// unsigned char \* outPut = malloc(16);

fscanf(f,"%s",&username);

//

if(strcmp(username,uname)==0)

{

fscanf(f,"%s",s.userpass);

unsigned char pass[10];

// printf("fine \n");

printf("Enter password: ");

char ch;

i=0;

while(1)

{

ch=getch();

if(ch=='\r')

break;

pass[i++]=ch;

ch='\*';

printf("%c",ch);

}

// scanf("%s",pass)2;

if(strcmp(pass,s.userpass)==0) //pass match printing details and menu card

{

struct username u;

fscanf(f,"%s %s %lld %s %s %lld \n",&u.personNamef,&u.personNamel,&u.phoneNo,&u.Email,&u.address);

printf("\nName: %s %s \nPhone No: %lld \nEmail: %s \nAddress: %s \n",u.personNamef,u.personNamel,u.phoneNo,u.Email,u.address);

int i;

for(i=0;i<50;i++)

printf("~-");

printf("\n");

menu();

break;

}

else

{

printf("\n\n ---------------\n");

printf(" |Wrong password| \n");

printf(" ---------------\n");

break;

}

}

else

{

fscanf(f, "%\*[^\n]\n");

}

if(feof(f))

{

printf("\n\n ------------------\n");

printf(" |Invalid Username| \n");

printf(" ------------------\n");

break;

}

}

fclose(f);

}

void deliveryboy()

{

char location[25];

printf("\nEnter your Location: ");

scanf("%s",&location);

printf("\nDelivery Boy name: John\n\n");

map(location); //showing map

}

int main()

{

int n;

int i;

for(i=0;i<50;i++)

printf("~-");

printf("\n");

printf("\t\tWelcome \n");

for(i=0;i<50;i++)

printf("~-");

printf("\n");

while(1)

{

printf("\nNEW ACCOUNT press 1 \n----------------------------------------------------\nLOGIN press 2\n----------------------------------------------------\nFor exit\t\t Press 0\n----------------------------------------------------\n ");

printf("Choice: ");

scanf("%d",&n);

printf("----------------------------------------------------\n");

switch (n)

{

case 1:

{

createAcc();

break;

}

case 2:

{

login();

break;

}

case 0:

{

exit(1);

}

default :

{

printf("\n\n ---------------\n");

printf(" |Invalid Input| \n");

printf(" ---------------\n");

}

}

}

return 0;

}