YEAR OF EXAMINATION: 2022-2023

SUBJECT: COMPUTER SCIENCE UNIT 2

PROFICIENCY: ADVANCED

CANDIDATE NAMES AND NUMBERS:

JOSEANN BONEO- 1600270112

ANYA-xxxxxxxxxxxxx- 160027

PARIS xxxxxxx- 160027xxxx

SAAYA xxxxxxxxxxxx- 160027xxxx

CENTRE NUMBER: 160027

CENTRE NAME: HOLY NAME CONVENT SECONDARY SCHOOL, PORT-OF-SPAIN

TEACHER'S NAME: MR. MICHAEL xxxxxx

Table of Contents

Page #

Specifications and Requirements

Problem Definition

Holy Name Convent is a prestige all girl secondary school founded by French Dominican nuns in 1982. The school strives to offer young Trinidadian women an opportunity to a better level of education, with a wide range of subjects to ensure each girl can pursue the job of their dreams. Holy Name Convent is a government assisted school, meaning that though they do receive money from the government it is not enough to ensure the proper upkeeping of the building. To pay for the maintenance of air conditioning or refurbishing classrooms, Holy Name Convent has an annual bazaar each year. This bazaar allows the school to make a large amount of money by providing different stalls of entertainment and food and drink on sale. This fundraiser is necessary as it not only produces a means of income for the school, but also allows for a spirit of comradery between students. Because of its importance, the fundraiser must run as smoothly as possible. However, the use of outdated manual systems of registration and financial keeping prohibits this process.

The outdated manual system requires users to edit, add, delete or calculate data using sheets of paper. While this system works it is not efficient because information can be easily misplaced, and human error is more likely to occur due to fatigue from manually entering the information. Our IA hopes to eliminate all errors that can potentially affect the administration of the event, those working in stalls and the customers who have come to enjoy themselves.

Firstly, administration would experience a lot of issues with a manual system as it is inefficient to book all the stall vendors by hand. The digitizing of the bazaar would allow administration to monitor the functionality of the system between customers and vendors without physically interacting with them. Additionally, instead of coming in school to sign up for what stall they wish to do, and what room they wish to occupy, vendors can fill this information out online. This would reduce the stress on both administration and vendors, make sure that overbooking or under booking does not take place and prevent time wastage as having all vendors come in can cause long lines and extended periods of waiting. Furthermore, it reduces stress on administration as they do not have to constantly keep track of the event planning. Simply making sure the site is running and that stalls are being filled properly will be adequate enough, leaving them energized for the actual bazaar day where they can be prepared for any mishaps.

Secondly, if our IA is implemented to create an automatic system for the bazaar, administration can then have access to detailed reports and analytics. Meaning administrators would be able to keep account of all the tickets bought by customers. This can then ensure the proper tracking of money and how much profit was made. Using an excel spreadsheet will allow administration to calculate sums using the already made functions in the program. It takes stress off administration after such a long event and is less time-consuming than the manual system. While the manual system allows for administration to check these things, human error is more likely to occur when dealing with such large numbers and administration can quickly become frustrated, tired or overwhelmed making the task seem more daunting than it is.

Thirdly, the use of an automated system will ensure that administration can effortlessly track the event and keep everyone, both staff and vendors, updated. Using their hand-held devices, administration should be able to see everything that is happening with the event in terms of planning; which vendors have signed up, entertainment and such. In a manual system they would use paper to keep track of such information which can easily become disorganized and frustrating. The online system should have a simple, sleek and easy to understand layout which allows them to view all information efficiently. Both systems use email to update users of any changes, however the manual is also likely to use phone calls, which can be time-consuming.

Lastly, by switching to an online bazaar system, you put less people at risk of both fatigue and illness. While Covid 19 may not be taken as seriously because the severity of the situation has declined significantly, it is still a very real problem. In having an online system, face-to-face contact is minimal, not only ensuring that key members of administration do not fall ill before such an important event, but also prevent fatigue or burnout from happening so early. Administrations can work easily from home or in their office without constantly having to interact with people in person, which can be very draining.

Techniques of Analysis

To solve the problem, a variety of techniques were used to provide clear and concise data to aid in the understanding of the actual problem being faced at Holy Name Convent. This allows for a specific solution to be created for the problem. These techniques included a questionnaire, an observation and a review of documents.

Questionnaire- a series of closed ended questions separated into sections depending on the category of persons involved in the bazaar which were the school's administration, past customer and vendors.

Observation- Observing the current system in practice and collecting the data in an observation table based on scenarios leading up to the day of the bazaar.

Review of Documents- Obtaining current documentation given to customers and vendors by the school's administration and examining how it is formatted.

The questionnaire was sent out to members of the school's administration, customers and vendors to get their feedback on the smoothness of the current bazaar management system. It comprised of 6 questions for the customers, 7 questions for the vendors and 8 questions for the administrators. All respondents returned the completed questionnaire in a timely manner. As for the observation, the table was crafted using data collected from students who observed the scenarios leading up to the bazaar. Lastly, the review of documents was conducted by obtaining receipts from a vendor and a customer to examine how it was formatted and if improvements could be made.

Questionnaire

CUSTOMERS QUESTIONS:

1. How did you find out about the bazaar?

- Online
- From a student
- From a teacher
- From a parent

2. How did you purchase your ticket?

- Online
- From a student
- From a teacher
- From a parent
- At the school on the day
-

3. How long did you wait to collect your ticket?

- A few days
- A week
- Immediately after purchase

4. What would you rate your overall experience in purchasing a ticket?

   1-5 scale

   1- Very dissatisfied, 5- very satisfied

5. Prior to the event, were you given any knowledge as to which vendors would be present at the event?

- Yes
- no

6. In the future, would you like to have prior knowledge about the vendors as well as their inventory?

- Yes
- no

VENDOR QUESTIONS:

7. How did you find out about this event?

- Online
- From a student
- From a teacher
- From a parent

8. How did you register your business for this event?

- Contact the school in person
- Through a pta member
- Online (school website)

9. Were you able to choose which stall you wanted prior to the event?

- Yes
- no

10. If yes, how did you choose?

- Contacted the school administration
- Contacted a teacher
- Online (school website)
- Through a pta member

11. Were you able to showcase your inventory to registered customers prior to the event?

- Yes
- no

12. If no, would you have preferred to?

- Yes
- no

13. What would you rate your overall experience of registering and working at the event?

    1-5 scale

    1- Very dissatisfied, 5- very satisfied

ADMIN QUESTIONS:

14. How did you advertise the bazaar for customers?

- Instagram
- Facebook
- School website
- PTA
- Students
- Teachers
- Radio
- Banners

15. How did you advertise the bazaar for vendors?

- Instagram
- Facebook
- School website
- PTA
- Teachers
- Radio
- Banners

16. How did you keep track of ticket sales?

- Excel sheet
- Database
- Human kept handwritten list

17. Were vendors able to advertise their inventory prior to the event?

- Yes
- no

18. If yes, what opportunities did they have to advertise?

    Open ended…

19. How did you keep track of customer and vendor information?

- Excel sheet
- Database
- Human kept handwritten list
- Sorted report

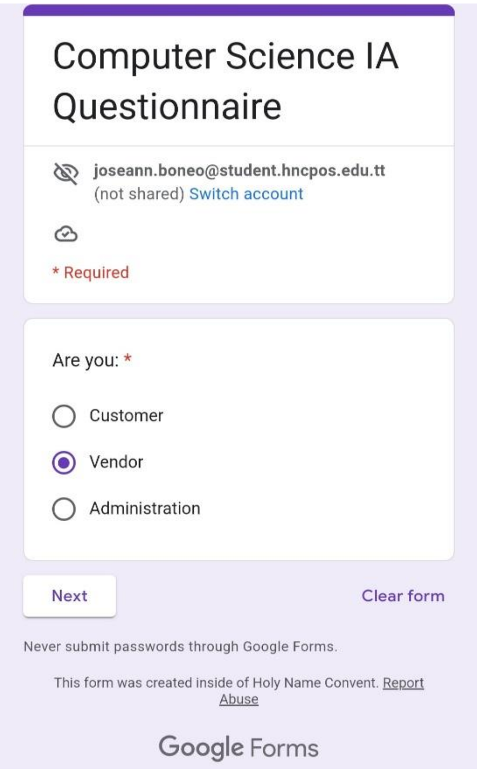
20. Would you prefer your customer and vendor registration to be an online system?

- Yes
- no

21. Was the overall management of registration of vendors and customers efficient?

- Yes
- no

Sample of a Completed Questionnaire



Computer Science IA Questionnaire

joseann.boneo@student.hncpos.edu.tt
(not shared) Switch account

* Required

Are you: *

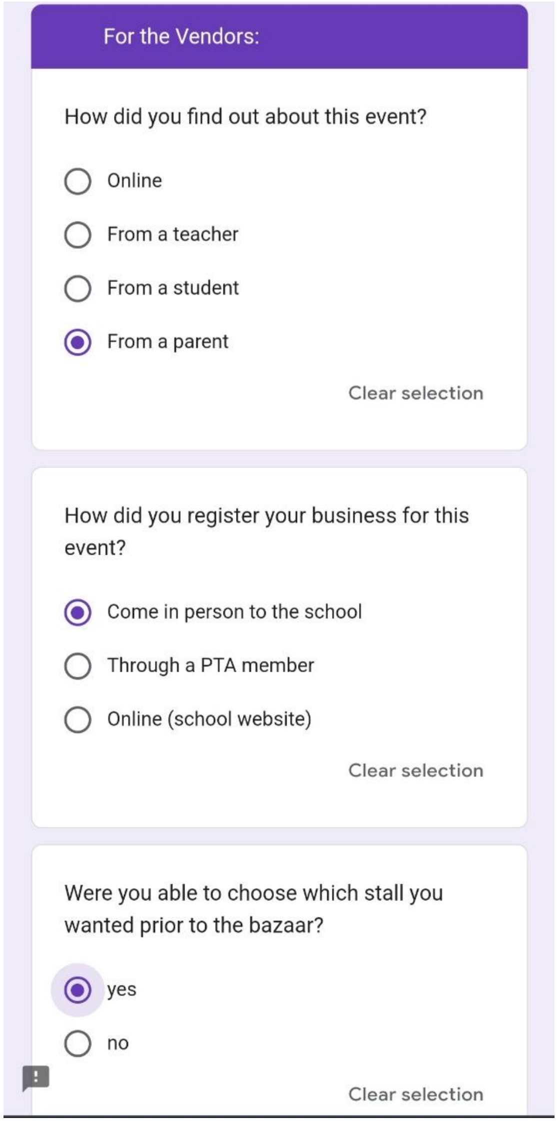
○ Customer
● Vendor
○ Administration

Next   Clear form

Never submit passwords through Google Forms.

This form was created inside of Holy Name Convent. Report Abuse

Google Forms



For the Vendors:

How did you find out about this event?

○ Online
○ From a teacher
○ From a student
● From a parent

Clear selection

How did you register your business for this event?

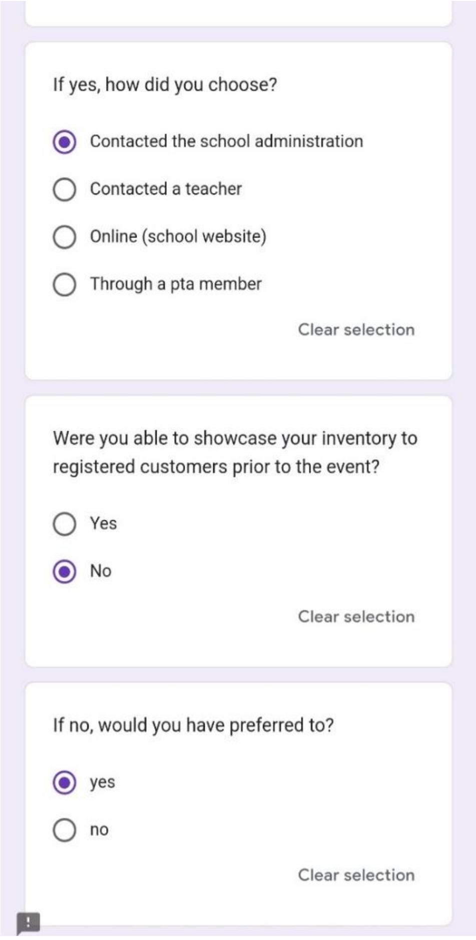
● Come in person to the school
○ Through a PTA member
○ Online (school website)

Clear selection

Were you able to choose which stall you wanted prior to the bazaar?

● yes
○ no

Clear selection



If yes, how did you choose?

● Contacted the school administration

○ Contacted a teacher

○ Online (school website)

○ Through a pta member

Were you able to showcase your inventory to registered customers prior to the event?

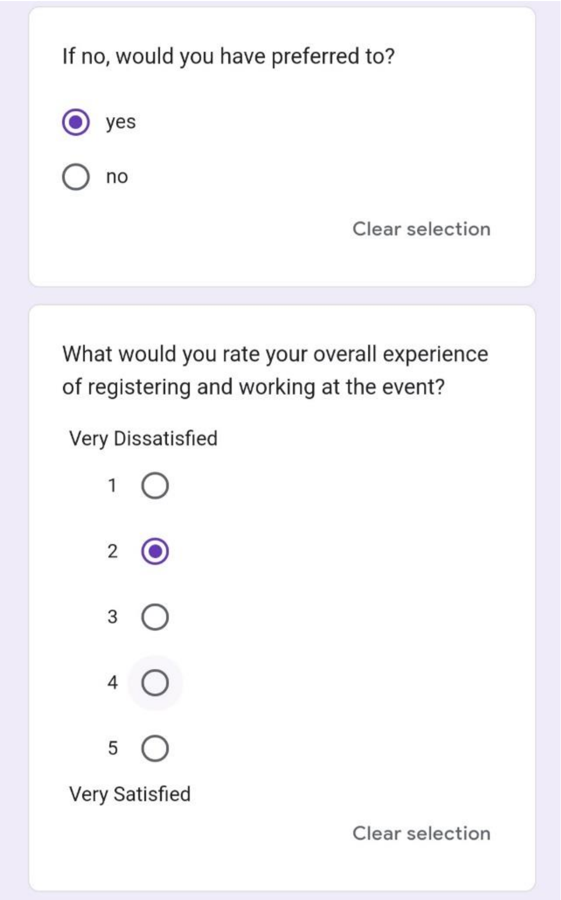
○ Yes

● No

If no, would you have preferred to?

● yes

○ no



If no, would you have preferred to?

● yes

○ no

What would you rate your overall experience of registering and working at the event?

Very Dissatisfied

1 ○

2 ●

3 ○

4 ○

5 ○

Very Satisfied

Questionnaire Charts

Customer Responses

Chart 1

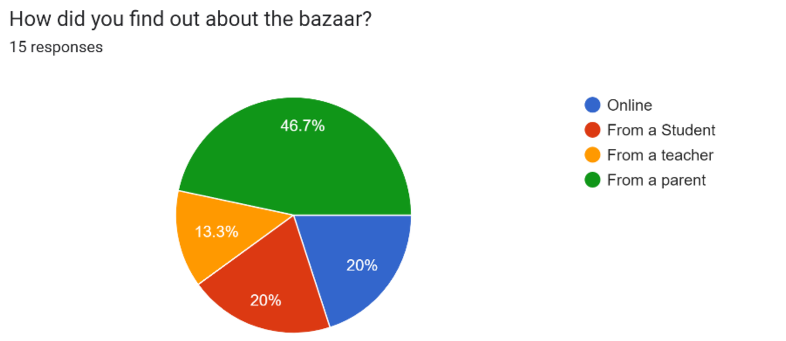


Chart 1 depicts the various replies from customers in response to how they found out about the bazaar. 46.7% of the respondents indicated that they found out from a parent, 20% found out online, 20% through a student and 13.3% found out through a teacher.

Chart 2

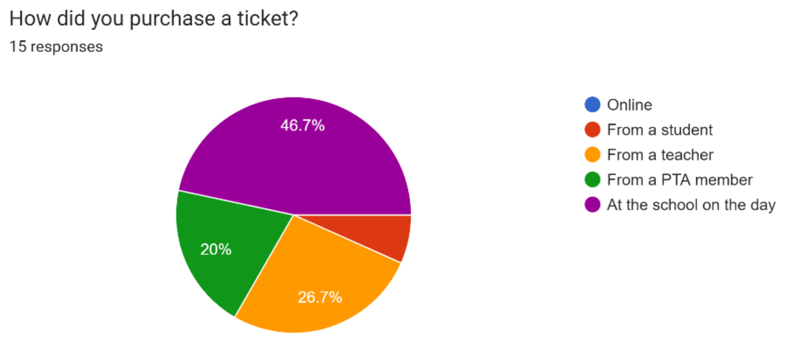


Chart 2 indicates how the customers purchased a ticket for the bazaar. 26.7% bought their tickets from a teacher, 20% bought from a PTA member, 46.7% bought at the school on the day and the remaining 6.6% purchased their tickets from a student.

Chart 3

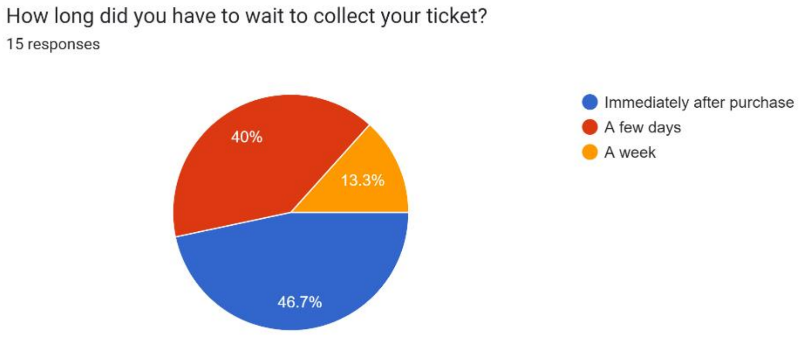


Chart 3 shows how long the customers had to wait to receive their tickets after purchasing it. 40% had to wait a few days, 13.3% had to wait a week and 40% received their tickets immediately after purchase.

Chart 4

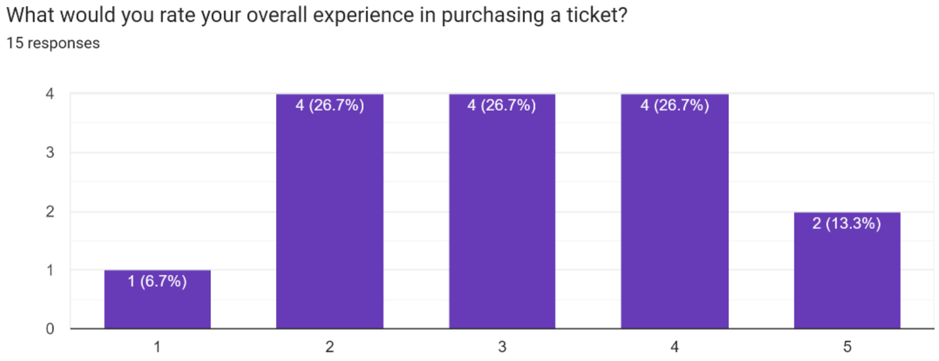


Chart 4 depicts the customers’ ratings of the overall experience of purchasing a ticket on a scale of 1 to 5, very dissatisfied to very satisfied. 6.7% of the respondents rated the experience a 1, 26.7% rated it a 2, 26.7% rated it a 3, 26.7% rated it a 4 and finally, 13.3% gave it an outstanding rating of a 5.

Vendor Responses

Chart 1

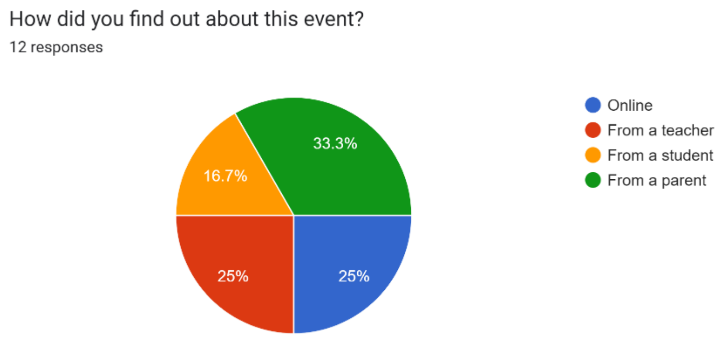


Chart 1 indicates how the vendors found out about the bazaar. 33.3% found out from a parent, 25% found out online, 25% found out from a teacher and the remaining 16.7% found out from a student.

Chart 2

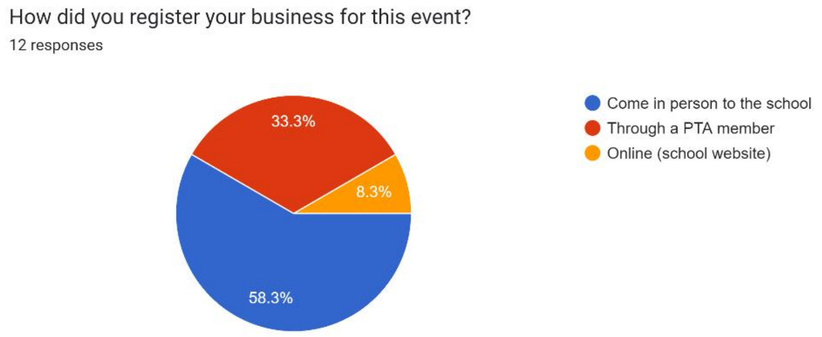


Chart 2 depicts the various ways that the vendors registered for the bazaar. 58.3% came in person to the school, 33.3% registered via a PTA member and 8.3% registered on the school website.

Chart 3

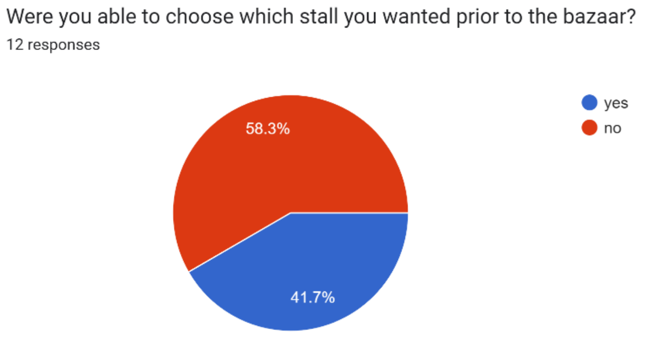


Chart 3 shows whether the vendors were able to choose their stall prior to the bazaar. 58.3% of the vendors were not able to choose their stall whereas 41.7% were able to choose.

Chart 4

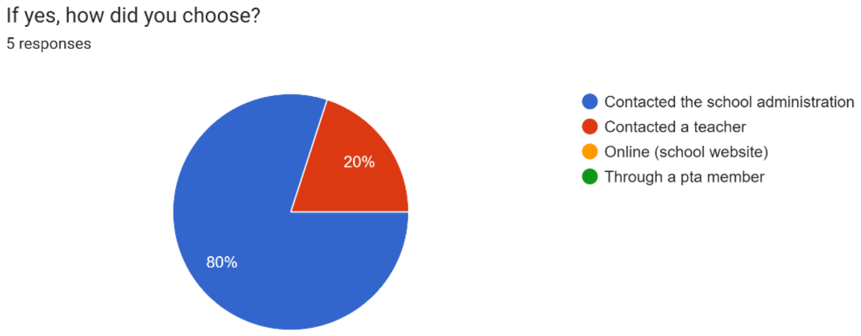


Chart 4 indicates how the vendors who were able to choose their stall chose. 80% contacted the school administration to choose whereas 20% contacted a teacher to choose.

Admin Responses

Chart 1

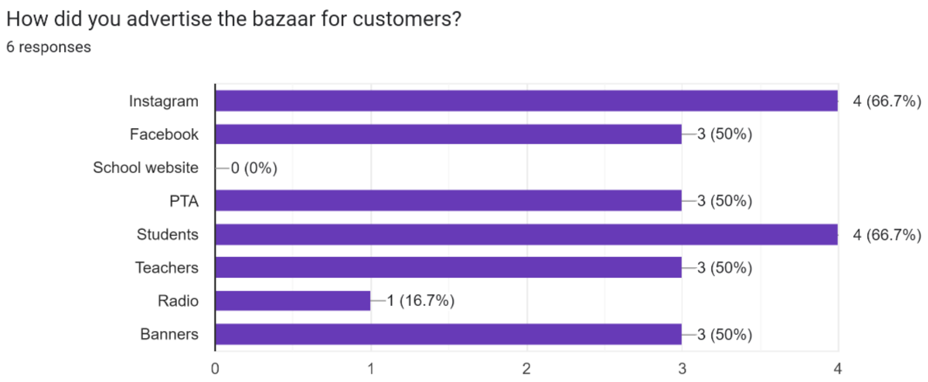


Chart 1 depicts how the school administration advertised the bazaar. They used a multitude of platforms such as Instagram, Facebook, Radio, Banners as well as through word of mouth by students, teachers and PTA members.

Chart 2

Chart 3

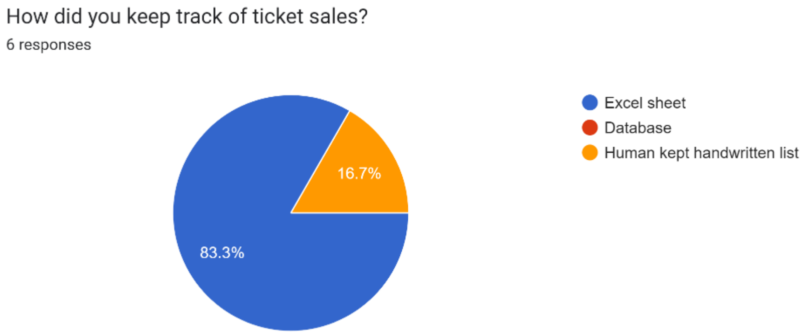
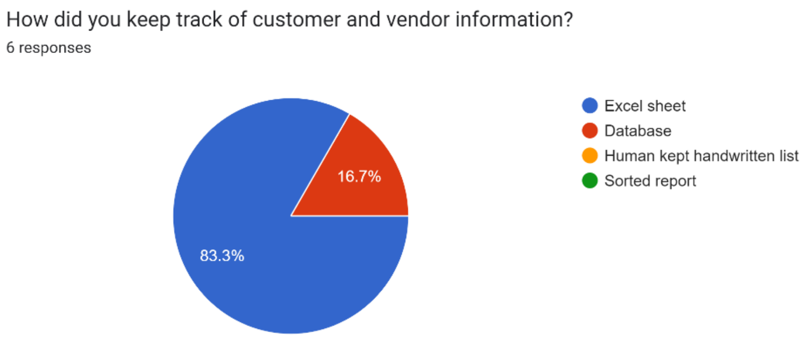


Chart 3 depicts the various ways that the school administration kept track of the ticket sales. They used both excel sheets and humans kept handwritten lists.

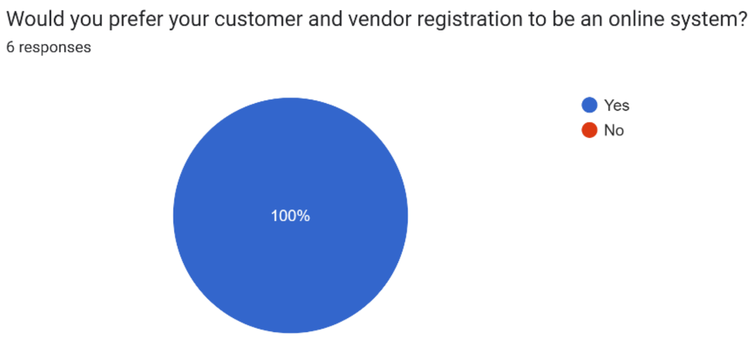
Chart 4


How did you keep track of customer and vendor information?

6 responses

Chart 4 indicates how the school administration kept track of both the customer and vendor information. To do this they made use of excel sheets and databases.

Chart 5


Would you prefer your customer and vendor registration to be an online system?

6 responses

Chart 5 shows whether the administration would prefer the customer and vendor registration to be an online system. It proved to be a unanimous yes.

Observation

Observation	Week 1	Week 2	Week 3	Day of
How many people were calling the school's administration to enquire about the bazaar? (customers/vendors)	30	18	29	0
How many people came into the office to purchase a ticket?	41	27	19	0
Average time taken to complete transaction (minutes)	5	6	7	6
Number of people seen walking out with a receipt or a ticket.	34	27	11	0
How many people came into the office to register as a vendor?	10	4	6	0
Average time taken to complete transaction (minutes)	15	19	16	0
Average time each person				10

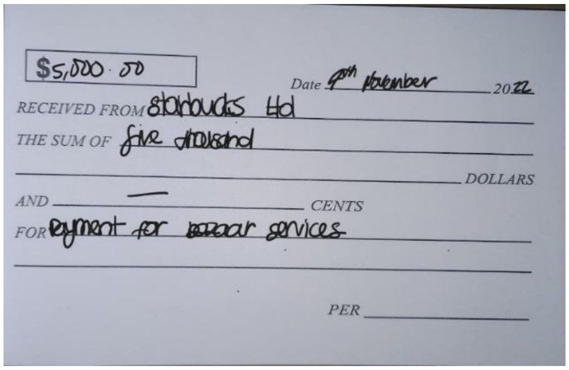
spent at 1 stall (minutes) (day of)				
Average number of people waiting in the line at 1 stall (day of)				10
Number of vendors occupying the stalls (including their assistance). (Day of)				53

The observation table above depicts the observances of the students of Holy Name Convent, Port of Spain regarding the ticket sales and vendors comings and goings in the weeks leading up to the bazaar as well as on the bazaar day itself. Regarding the incoming calls about the bazaar, the school received approximately 30 calls 3 weeks away from the event, 18 calls 2 weeks away and 29 calls 1 week away. When it came to purchasing tickets in person, 41 people came into the office to buy tickets 3 weeks prior to the event, 27 people 2 weeks prior and 19 people 1 week prior. The average time taken to complete these transactions varied depending on the week; 3 weeks before the event, the transactions took approximately 5 minutes to complete, in week 2, 6 minutes, in week 3, 7 minutes and on the day itself, the transactions were completed in around 6 minutes. Usually, when a payment is made, a receipt is given, hence, an individual who came in person to purchase a ticket for the bazaar should have received a receipt. Thus, 34 people were seen leaving the office with a receipt or a ticket as their receipt 3 weeks away from the bazaar, 27 people left the office with a form of receipt 2 weeks away from the bazaar and 11 people were seen leaving the office with a receipt or ticket 1 week away from the bazaar. Many of the vendors contacted people associated with the school to register for the event, however, 3 weeks away from the event, 10 people came in person to register, 4 people came 2 weeks away from the event and 6 people came 1 week out. The average times to complete this transaction, again, varied depending on the week; with 3 weeks left to the event, the transactions took approximately 15 minutes, with 2 weeks left, 19 minutes and with 1 week left, 16 minutes. The students also paid careful attention to the happenings on the day of the bazaar with respect to the time spent at each stall, approximately 10 minutes, approximately 10 people waiting in line at each stall as well as approximately 53 vendors occupying stalls at the event.

Review of Documents:

Document 1

Document 1. is a copy of a handwritten receipt from the books kept by the school bursar. The receipt is visibly illegible, and many important details are indecipherable.



$5,000.00

Date 4th November 2022

RECEIVED FROM Starbucks Ltd

THE SUM OF five thousand

DOLLARS

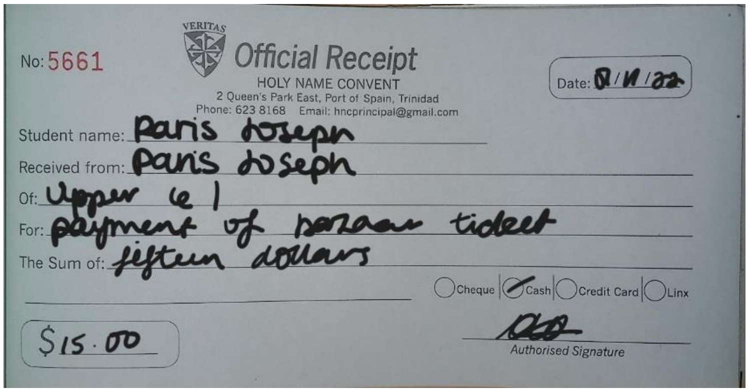
AND — CENTS

FOR payment for bazaar services

PER

Document 2

Document 2. shows the copy of a receipt given to a student at the school in return for payment of a bazaar ticket. Notably, the handwriting on the receipt is illegible and untidy.



No: 5661

VERITAS

Official Receipt

HOLY NAME CONVENT
2 Queen's Park East, Port of Spain, Trinidad
Phone: 623 8168 Email: hncprincipal@gmail.com

Date: [illegible]/11/22

Student name: Paris Joseph

Received from: Paris Joseph

Of: Upper 6 1

For: payment of bazaar ticket

The Sum of: fifteen dollars

○ Cheque ☑ Cash ○ Credit Card ○ Linx

$15.00

Authorised Signature

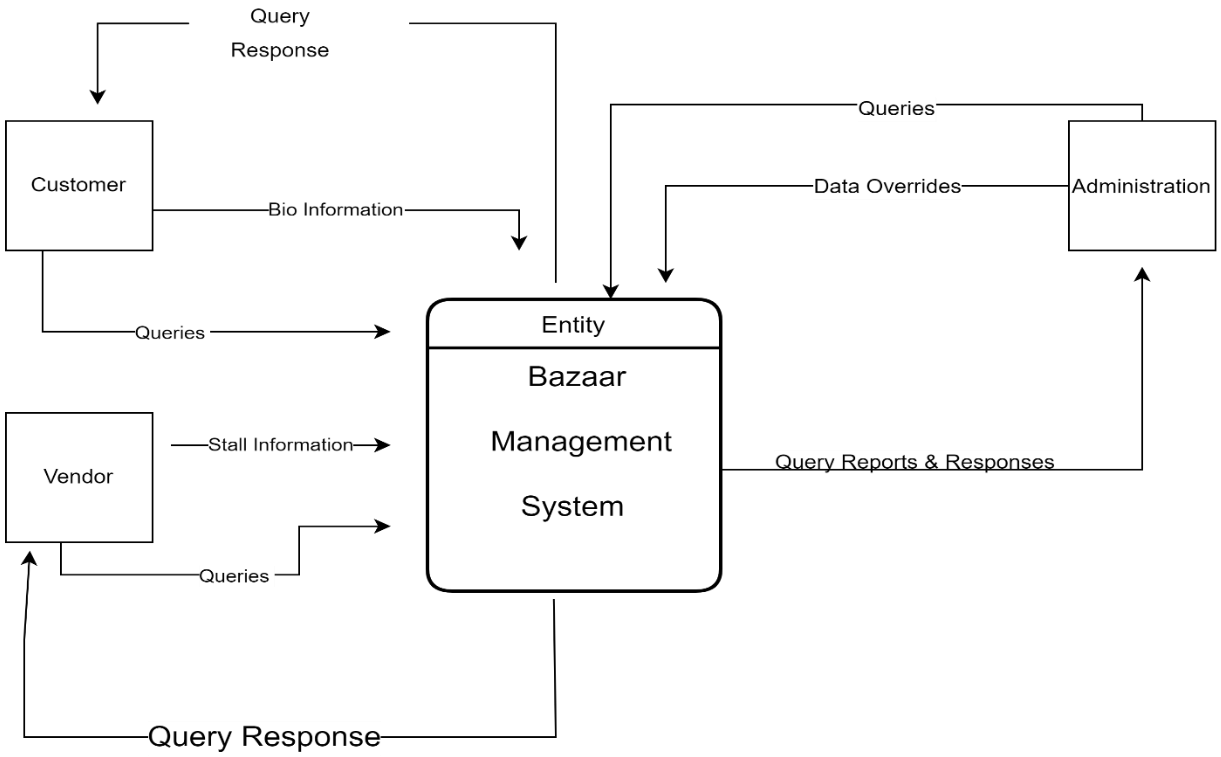
Final Analysis

The questionnaire was sent out to members of the school's administration, customers and vendors to get their feedback on the smoothness of the current bazaar management system. It comprised of 6 questions for the customers, 7 questions for the vendors and 8 questions for the administrators. All respondents returned the completed questionnaire in a timely manner and there answers highlighted a issues with the dissemination and accessibility of information concerning the event. As for the observation, the table was crafted using data collected from students who observed the scenarios leading up to the bazaar. Lastly, the review of documents was conducted by obtaining receipts from a vendor and a customer to examine how it was formatted and if improvements could be made for reports.

Use of Data Flow Data Diagrams and Entity Relationship Diagram

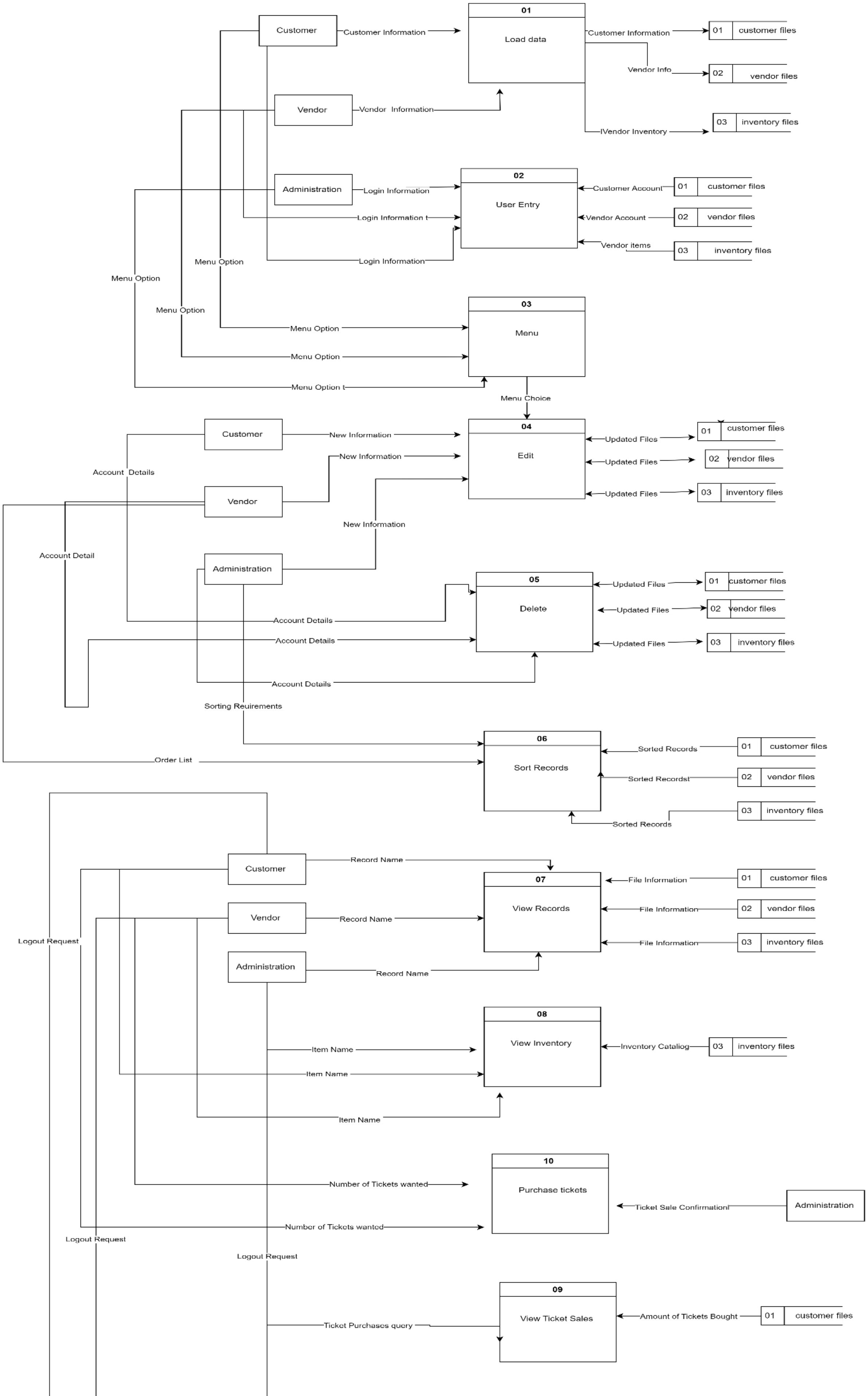
Context Level Data Flow Diagram

The context level diagram shows the basic operations of the Bazaar Management System. Customers enter their information or queries into the system and the system returns the query responses. Vendors enter stall information or queries, and the system returns query response. Administration sends queries and data overrides and receives data responses and queries.



Level 1 Data Flow Diagram

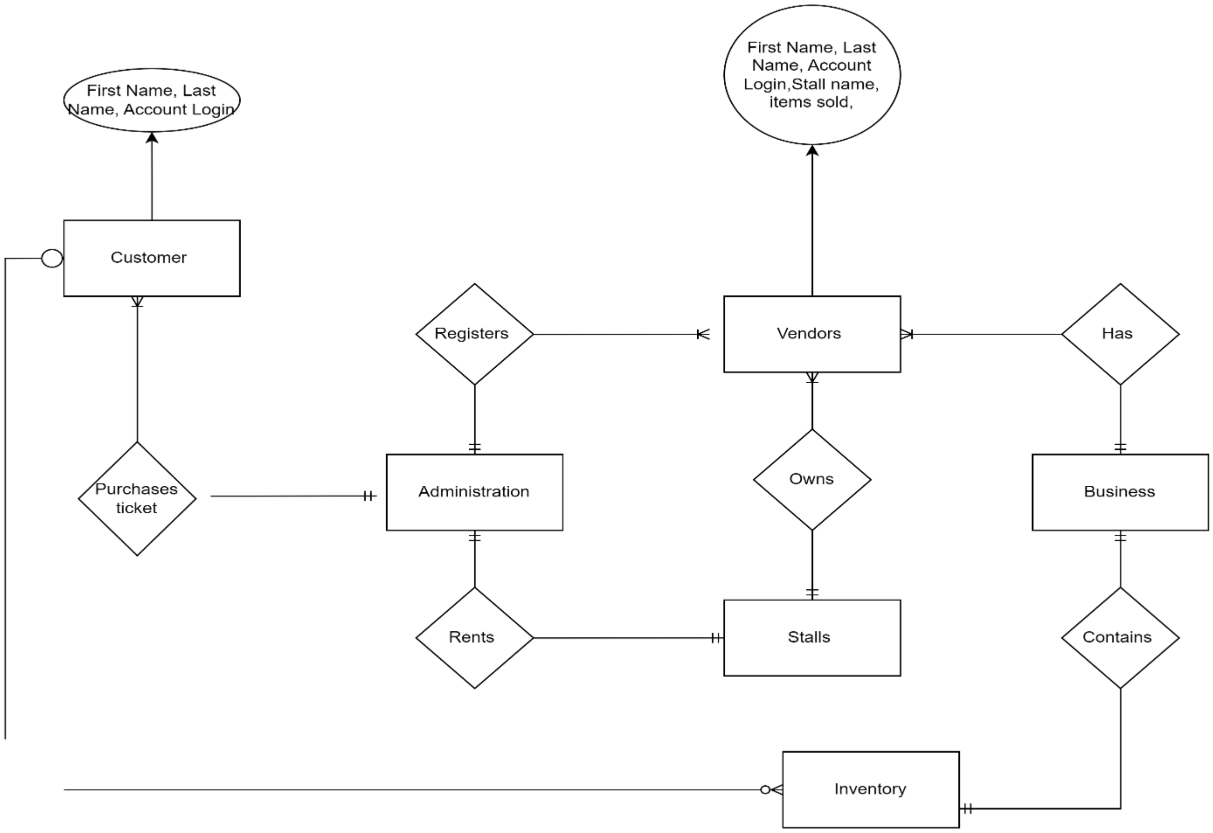
A data-flow diagram is a way of representing a flow of data through a process or a system. The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow — there are no decision rules and no loops.


Customer
Customer Information
01
Load data
Customer Information
01 customer files
Vendor Info
02 vendor files
03 inventory files
IVendor Inventory
Vendor
Vendor Information
Administration
Login Information
02
User Entry
Customer Account
01 customer files
Login Information t
Vendor Account
02 vendor files
Vendor items
03 inventory files
Login Information
Menu Option
Menu Option
Menu Option
03
Menu
Menu Option
Menu Option
Menu Option t
Menu Choice
Customer
New Information
04
Edit
Updated Files
01 customer files
Account Details
Vendor
New Information
Updated Files
02 vendor files
Updated Files
03 inventory files
New Information
Account Detail
Administration
05
Delete
Updated Files
01 customer files
Updated Files
02 vendor files
Account Details
Account Details
Updated Files
03 inventory files
Account Details
Sorting Reuirements
06
Sort Records
Sorted Records
01 customer files
Order List
Sorted Recordst
02 vendor files
03 inventory files
Sorted Records
Customer
Record Name
07
View Records
File Information
01 customer files
Logout Request
Vendor
Record Name
File Information
02 vendor files
File Information
03 inventory files
Administration
Record Name
08
View Inventory
Item Name
Inventory Catalog
03 inventory files
Item Name
Item Name
10
Purchase tickets
Number of Tickets wanted
Ticket Sale Confirmationl
Administration
Number of Tickets wanted
Logout Request
Logout Request
09
View Ticket Sales
Amount of Tickets Bought
01 customer files
Ticket Purchases query

Entity Relation Diagram

The Entity Relation Diagram is a graphical representation that depicts relationships among people, objects, places, concepts or events within a system. In our IA the customer can purchase a ticket from administration or view inventory. Vendors register their stalls through administration, owns stalls and have businesses that contain inventory. The administration rents stalls which the vendors own and sells multiple tickets to many customers.

Entity	Attributes
Customer	Customer ID, first Name, last Name, gender, contact, num tickets bought, password
Administration	password
Vendor	Vendor ID, first Name, last Name, contact, password
Stall	Stall ID, stall category
Business	Business name,
Inventory	Stall ID, num items, item Name


First Name, Last Name, Account Login
First Name, Last Name, Account Login,Stall name, items sold,
Customer
Registers
Vendors
Has
Purchases ticket
Administration
Owns
Business
Rents
Stalls
Contains
Inventory

Functional Requirements

1. The System should be able to **add**:
    a. A Vendor – A new record will be saved by the program in a text file in the vendor's personal record format when a potential vender enters their name, contact info, business name, stall category, stall location, and password. Another record would be saved by the program in a text file in the vendor's inventory record format when they enter their user ID and the various items in their inventory. Finally, the program would indicate that this function was completed by allowing the user to navigate the menu.
    b. A Customer – A new record will be saved by the program in a text file in the customer's personal record format when they enter their name, contact information and gender.
    The captured information for the vendor and customers will be stored in the appropriate data structures. Finally, the program would indicate that this function was completed by allowing the user to navigate the menu.
2. Customers and vendors would be able to **view** their personal information from their respective from the array of structure in the text file by using their user ID and linear searching the array for their record and displaying their information.
3. The customer and the school's administration will be able to **search** for a specific vendor by the desired vendor ID that they enter. A linear search would be used to identify the vendor and display their business and inventory information.
4. The customer will be able to purchase additional tickets by selecting the option in the menu and inputting the number of tickets they wish to purchase. The value entered would be saved in their records in the text file by searching the array of structures for their record.
5. The customer will be able to **edit** their information by using their user ID, search the customer text file to locate their information, the program would allow them to enter new information and store their new record array in the text file. For example, edit their name, contact information and ticket purchases. Finally, the program should print a message to indicate that this function was completed
6. The system should provide the administration with a variety of reports to **view**:

   a. Customer Reports- list of pre-registered customers or total ticket sales from the array of structure in the customer text file.
   b. Vendor- list of vendors or inventories of all vendors from the array of structure in the vendor text file.

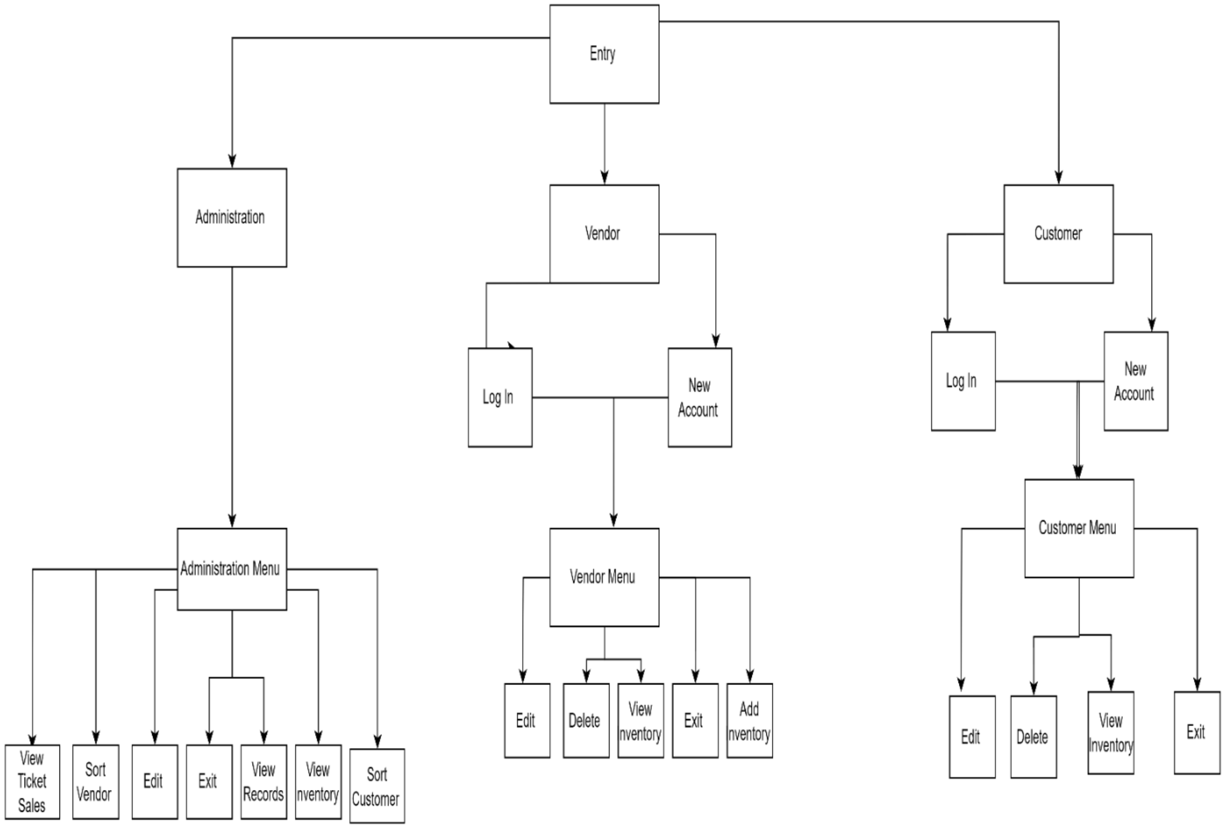
7. Administration should be able to **sort** the array of structure in either the customer or vendor text file using a bubble sort and display the respective information from the sort.
8. Administration should enter the user ID of a customer or vender to **search** the array of structure in the respective text file and display their personal information. Additionally, administration would have the option to display the inventory of the selected vendor.
9. Vendors should be able to input the items in their inventory then their items would be **added** to a text file with the in the vendor's inventory record format at the end the previous records. Finally, the program should print a message to indicate that this function was completed.
10. Vendors should be able to **delete** items from their inventory. Firstly, the program would use the user Id and searches through the array of structures for vendor records. Then they would enter the item they wish to remove by using a linear search to identify the item saved in the file's records and overwrite the item with all other items in the array of structures. Finally, the program should print a message to indicate that this function was completed.
11. Vendors should be able to **update** their inventory in the array of structure in the text file by entering the item which would be stored in their record array after their user ID is used to find their record array in the text file. Finally, the program should print a message to indicate that this function was completed
12. Administration will be able to **delete** a customer or vendor from the system by entering their customer or vendor's user ID, a linear search would be used to find the user in the respective file in the array of structures and overwrites the record will all the other records in the array. Finally, the program should print a message to indicate that this function was completed

Non-Functional Requirements

1) Security:
    a. Passwords for all the stakeholders
        i. Vendors
        ii. Customers
        iii. Administration
2) The system should back up all the active data stored in the data structures every thirty minutes
3) The system should be installed on a computer with a minimum of 4GB of RAM.
4) The system should provide an easy-to-use menu driven user interface for the vendors, customers and administration.
5) Within the code of the program, short notes are made by the programmers to assist future users in understanding the inner working of the program. These documentations offer description of the various functions and purposes of any section of the code
6) After three hours of training, administration should be able to easily navigate the system and errors experienced by experienced users should not exceed 3 per hour
7) The system shall be available 24/7 to all customers and vendors 3 months before the event.
8) Upon startup the system should be responsive and available to use within 3 seconds.
9) The programming will display an error message in the event of incorrect data type entry. These messages reduce the chance of incorrect data being entered and recorded.

Design Specifications

System Structure



This diagram shows how the menu will be structured from the user's perspective.

1. Entry- This initial menu prompts users to select their respective user type either admin, customer or vendor
2. Administration Log in- This function prompts this type of user to enter their password to view their respective menu options
3. Vendor Entry- This function asks vendors to select the appropriate user type, either a new vendor or an existing vendor.
4. Vendor Log In- This function asks vendors to enter their correct user ID and password.
5. New Vendor Account- This function registers a new vendor to the system by allowing them to enter their personal and business information.
6. Customer Entry- This function asks customers to select the appropriate user type, either a new customer or an existing customer.
7. Customer Log In- This function asks customers to enter their correct user ID and password.

8. New Customer Account- This function registers a new customer to the system by allowing them to enter their personal information.

Administration Menu

1. View Ticket Sales- This function allows admin to view all the customers that purchased a ticket with their total as well as see a grand total of all the sales.
2. Sort Vendor Records- This function uses a bubble sort to sort all the records of vendors stored in the system and details their information under labeled columns.
3. Sort Customer Records- This function uses a bubble sort to sort all the records of the customer stored in the system and details their information under labeled columns.
4. Delete Vendor Record- This function allows admin to permanently delete all the information associated with the selected vendor in the system.
5. Delete Customer Record- This function allows admin to permanently delete all the information associated with the selected customer in the system.
6. View Customer Record- This function uses a linear search to find the selected customer information in the system.
7. View Vendor record- This function uses a linear search to locate the personal information about the selected vendor in the system.
8. View Inventories- This function allows admin to view all the business and inventory details of all the vendors in the system under labeled columns.
9. Exit System- This function prompts the user with a goodbye message.

Customer Menu

1. View Customer Record- This function uses a linear search to find their customer information in the system based on their saved ID from logging into the system.
2. View Vendor record- This function uses a linear search to locate the personal information about the selected vendor in the system.
3. Edit Customer Record- This function allows the customer to edit their personal information in the system.
4. Purchase Ticket- This function lets customers enter the number of tickets they wish to purchase and provides the total amount to be paid.
5. Delete Customer Record- This function allows customers to permanently delete all the information associated with their ID in the system.
6. Exit System- This function prompts the user with a goodbye message.

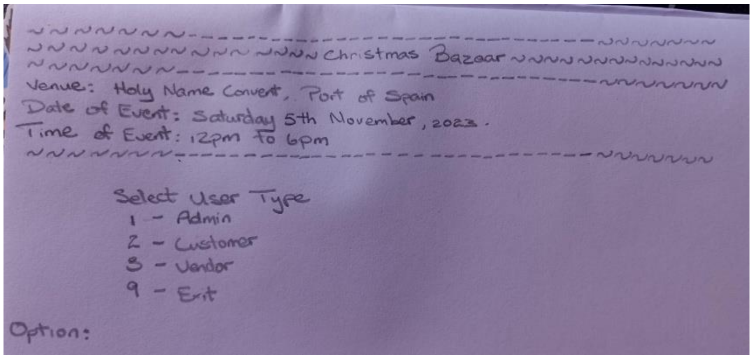
Vendor Menu

1. View Vendor record- This function uses a linear search to locate their personal information in the system based on their saved ID from logging into the system.
2. Edit Vendor Record- This function allows the vendor to edit their personal and stall information in the system.
3. Delete Vendors Record- This function allows vendors to permanently delete all the information associated with their ID in the system.
4. View Inventory- This function allows vendors to view their business's stall information and all the items uploaded to their inventory in the system
5. Add Inventory- This function allows the vendors to upload more items to their inventory in the system.
6. Exit System- This function prompts the user with a goodbye message.

User Interface Design

To ensure easy navigation of the system by customers, vendors and administration, a menu driven interface was used so that one menu leads to a further menu depending on the initial selected user type. The instruction for each menu is followed by a numbered list of options for users to choose from. This type of human computer interface ensures that users do not have to remember any set of commands, which makes the system have self-explanatory menu options for first time users.

Main Menu

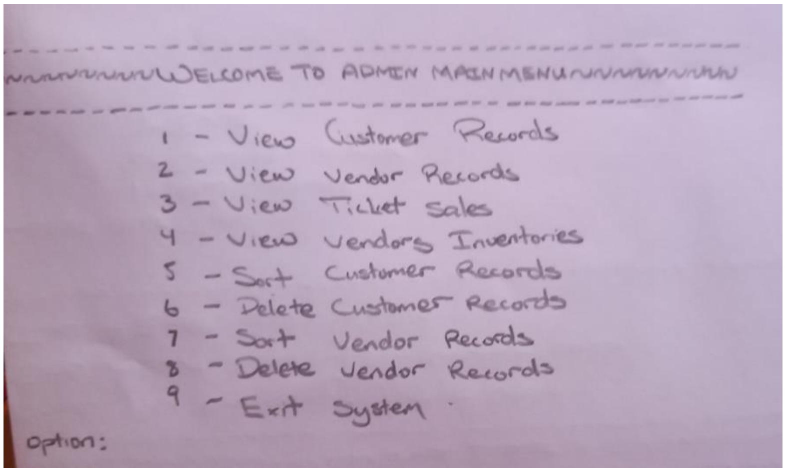


```
~~~~~~~~~---------------------------------------~~~~~~~~
~~~~~~~~~~~~~~~~~~~~ Christmas Bazaar ~~~~~~~~~~~~~~~~~~~
~~~~~~~~~---------------------------------------~~~~~~~~~
Venue: Holy Name Convent, Port of Spain
Date of Event: Saturday 5th November, 2023.
Time of Event: 12pm to 6pm
~~~~~~~~~---------------------------------------~~~~~~~~

        Select User Type
         1 - Admin
         2 - Customer
         3 - Vendor
         9 - Exit
Option:
```

The title of the database was separated using a line to clearly show the user that they are in the system.

Each user type option is clearly numbered to ensure legibility of the options for the user. A colon was used to indicate to the user where to make their choice.

Administration Menu



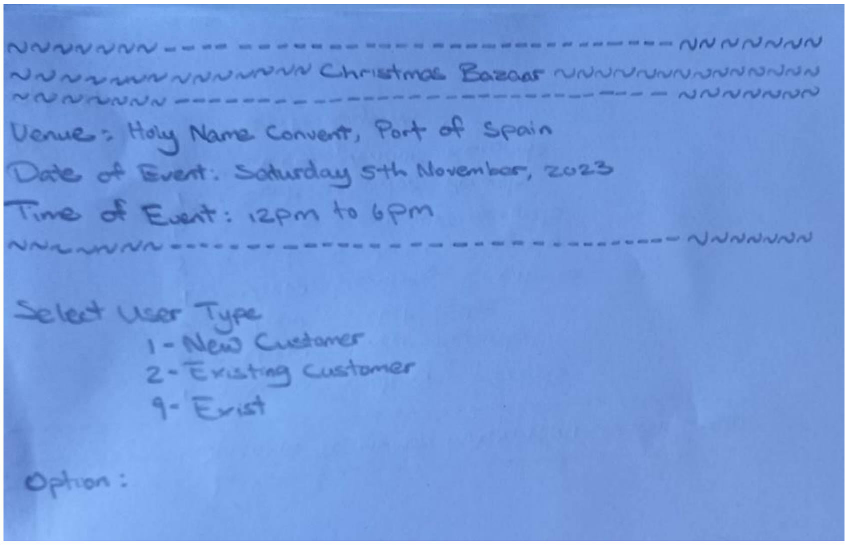
```
-------------------------------------------------
~~~~~~~~~WELCOME TO ADMIN MAIN MENU~~~~~~~~~~
-------------------------------------------------
            1 - View Customer Records
            2 - View Vendor Records
            3 - View Ticket Sales
            4 - View Vendors Inventories
            5 - Sort Customer Records
            6 - Delete Customer Records
            7 - Sort Vendor Records
            8 - Delete Vendor Records
            9 - Exit System .
option:
```

The tilde symbol was used to indicate that the administration main menu was chosen.

Each type of record is clearly numbered for the user to choose from.

A colon was used to indicate to the user where to make their choice.

Customer Type Selection Menu



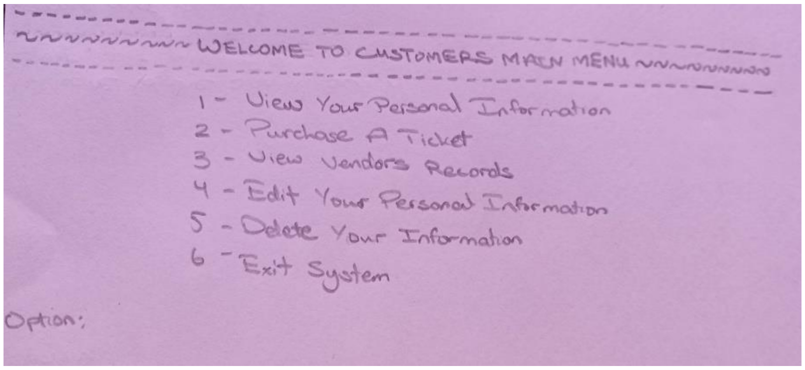
```
~~~~~~~------------------------------------~~~~~~~
~~~~~~~~~~~~~~~~ Christmas Bazaar ~~~~~~~~~~~~~~~~
~~~~~~~------------------------------------~~~~~~~
Venue: Holy Name Convent, Port of Spain
Date of Event: Saturday 5th November, 2023
Time of Event: 12pm to 6pm
~~~~~~~------------------------------------~~~~~~~

Select User Type
         1- New Customer
         2- Existing Customer
         9- Exist

Option :
```

The title of the database was separated with a line to show the user that they are in the system.

Each customer type is numbered to allow the ease of user choice.

A colon was used to indicate to the user where to make their choice.

Customer Menu

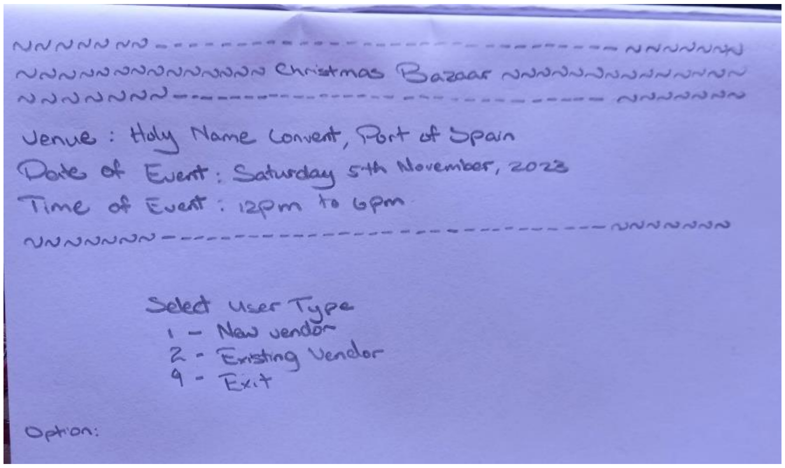


```
------------------------------------------------------------
~~~~~~~~~ WELCOME TO CUSTOMERS MAIN MENU ~~~~~~~~~
------------------------------------------------------------
               1 - View Your Personal Information
               2 - Purchase A Ticket
               3 - View Vendors Records
               4 - Edit Your Personal Information
               5 - Delete Your Information
               6 - Exit System
Option:
```

The tilde symbol was used again to show that the customer's main menu was selected. The option choices are numbered for the customers easy understanding.

A colon was used to indicate to the user where to make their choice.

Vendor Type Selection Menu



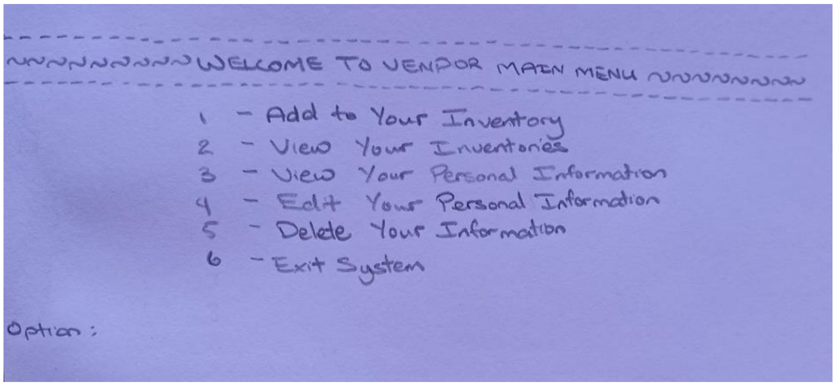
```
~~~~~~~------------------------------------------~~~~~~~
~~~~~~~~~~~~~~~ Christmas Bazaar ~~~~~~~~~~~~~~~
~~~~~~~------------------------------------------~~~~~~~
Venue : Holy Name Convent, Port of Spain
Date of Event : Saturday 5th November, 2023
Time of Event : 12pm to 6pm
~~~~~~~------------------------------------------~~~~~~~

          Select User Type
            1 - New vendor
            2 - Existing Vendor
            9 - Exit

Option:
```

The title of the database was separated with a line to show the user that they are in the system.

Each option is labeled with a number so that the user can make their choice easily.

A colon was used to indicate to the user where to make their choice.

Vendor Menu



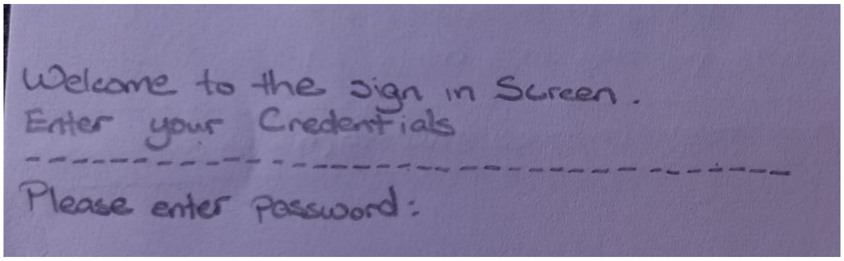
```
----------------------------------------------------------------
~~~~~~~~~WELCOME TO VENDOR MAIN MENU ~~~~~~~~~
----------------------------------------------------------------
            1  - Add to Your Inventory
            2  - View  Your  Inventories
            3  - View  Your  Personal  Information
            4  - Edit  Your  Personal  Information
            5  - Delete  Your  Information
            6  - Exit  System

Option :
```

The tilde symbol was used to depict the vendor's main menu being chosen.

The various choices for the vendors were organized numerically.

A colon was used to indicate to the user where to make their choice.

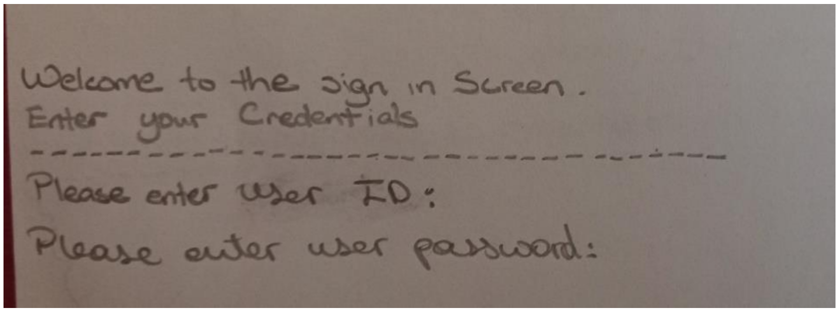
Administration Entry Screen



```
Welcome to the sign in Screen.
Enter your Credentials
-----------------------------------------------
Please enter password:
```

The title of the database was separated with a line to show the user that they are in the system. A colon was used to indicate to the user where to make their choice.

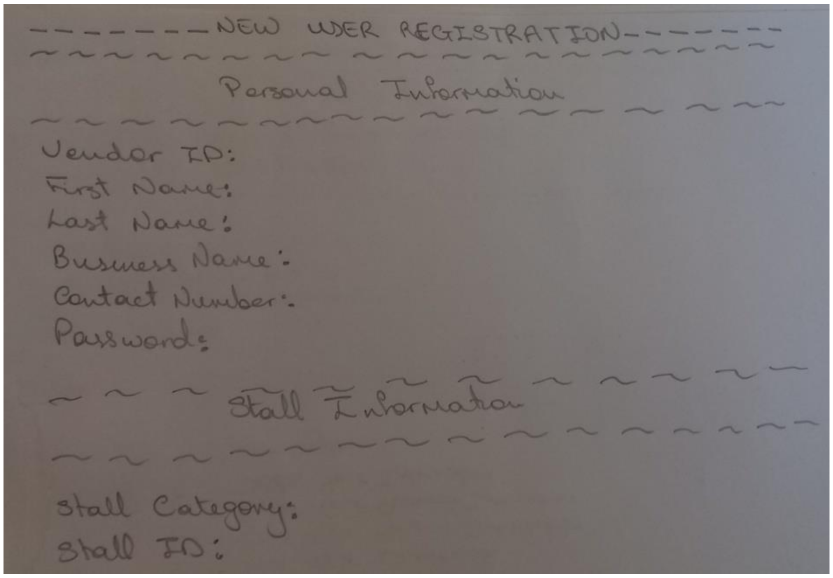
Customer/Vendor Entry Screen



```
Welcome to the sign in Screen.
Enter your Credentials
-----------------------------------------------
Please enter user ID:
Please enter user password:
```

The title of the database was separated with a line to show the user that they are in the system. Colons were used to indicate to the user where to make their choice.

Vendor Sign Up Screen



NEW USER REGISTRATION

Personal Information

Vendor ID:

First Name:

Last Name:

Business Name:

Contact Number:

Password:

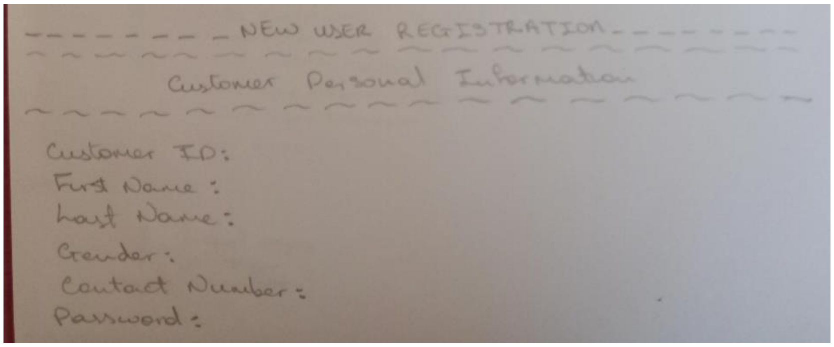
Stall Information

Stall Category:

Stall ID:

The title of the database was separated with a line to show the user that they are in the system. Colons were used to indicate to the user where to make their choice.

Customer Sign Up Screen



NEW USER REGISTRATION

Customer Personal Information

Customer ID:

First Name :

Last Name:

Gender:

Contact Number:

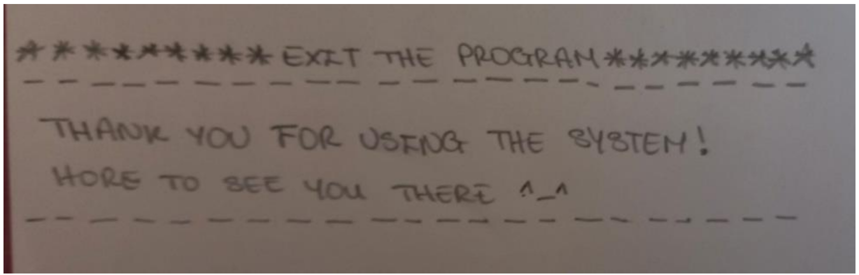
Password:

The title of the database was separated with a line to show the user that they are in the system. Colons were used to indicate to the user where to make their choice.

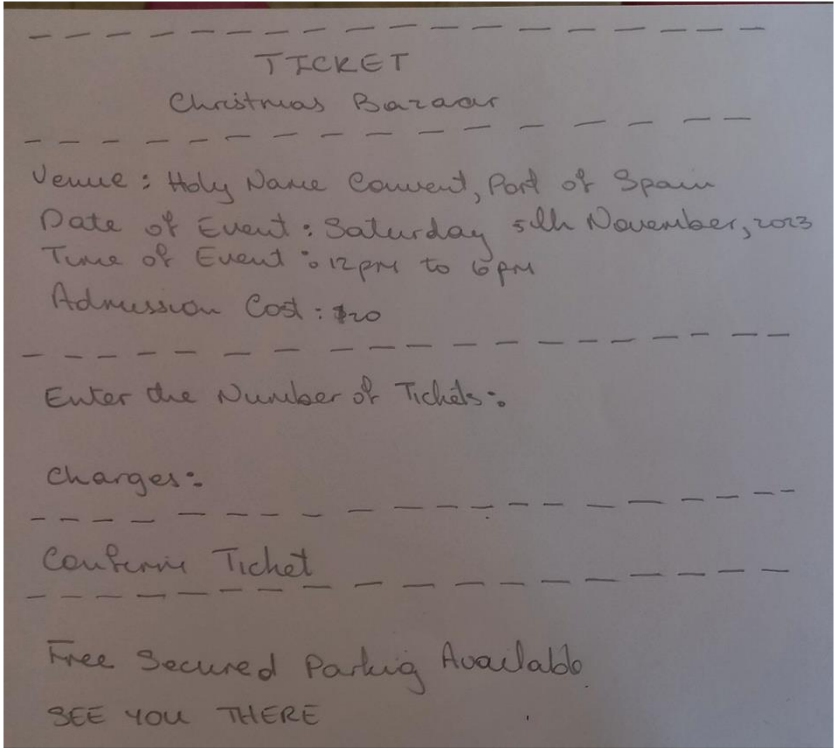
Customer Ticket Purchasing Screen

The title of the database was separated with a line to show the user that they are in the system. Colons were used to indicate to the user where to make their choice.

Exit Message



```
*********EXIT THE PROGRAM*********
-----------------------------------

THANK YOU FOR USING THE SYSTEM!
HORE TO SEE YOU THERE ^_^
-----------------------------------
```

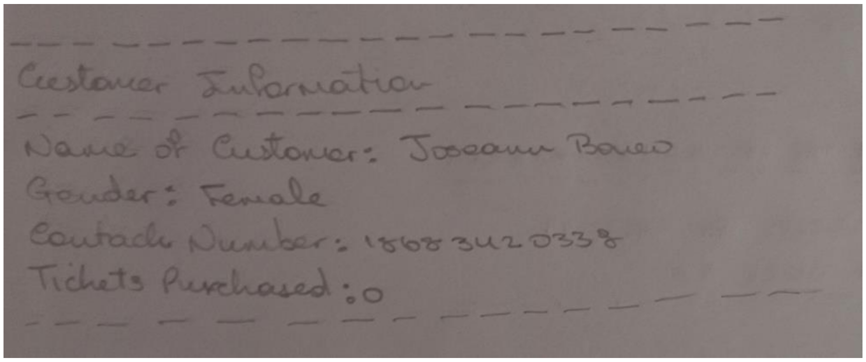


```
-----------------------------------
              TICKET
         Christmas Bazaar
-----------------------------------
Venue : Holy Name Convent, Port of Spain
Date of Event : Saturday 5th November, 2023
Time of Event : 12pm to 6pm
Admission Cost : $20
-----------------------------------

Enter the Number of Tickets :

Charges :
-----------------------------------
Confirm Ticket
-----------------------------------

Free Secured Parking Available
SEE YOU THERE
```

A nice message was left for the user at the end using various special characters to appeal to user as their usage of the system has come to an end.

Report Design

Searched Customer Information Report

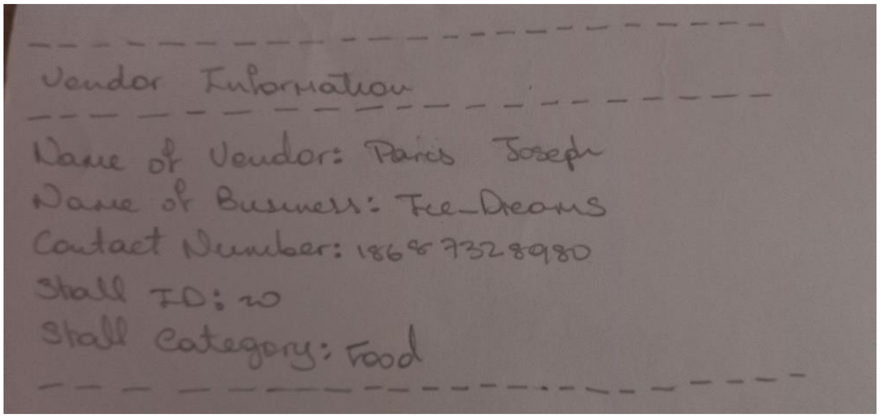


```
------------------------------------------
Customer Information
------------------------------------------
Name of Customer: Joseann Bowen
Gender: Female
Contact Number: 18683420338
Tickets Purchased: 0
------------------------------------------
```

Tilde symbols were used to show the user that the customer information report has been chosen.

Colons were used to show the user where to put the information.

Searched Vendor Information Report

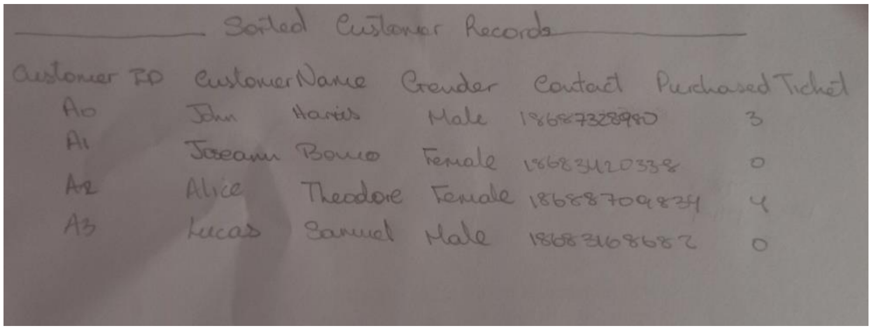


```
------------------------------------------
Vendor Information
------------------------------------------
Name of Vendor: Paris Joseph
Name of Business: Ice-Dreams
Contact Number: 18687328980
Stall ID: 20
Stall Category: Food
------------------------------------------
```

The title of the report was separated from the system using a line.

Colons were used to show the user the information requested.

Sorted Customer Records Report

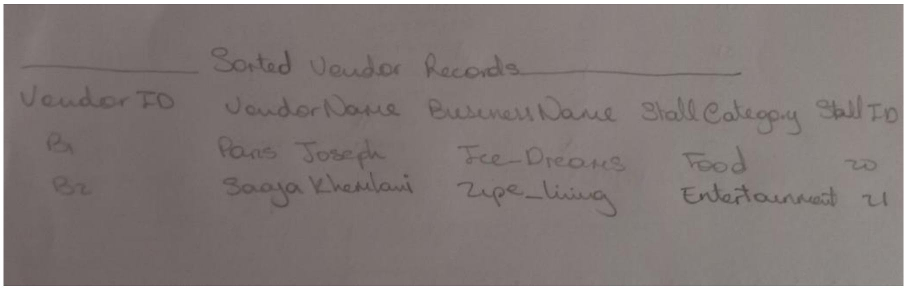


Sorted Customer Records

Customer ID	Customer Name		Gender	Contact	Purchased Ticket
A0	John	Harries	Male	18687328980	3
A1	Joseann	Bourro	Female	18683420338	0
A2	Alice	Theodore	Female	18688709834	4
A3	Lucas	Samuel	Male	18683168682	0

Headings were used to show the individual customer records that were sorted.

Sorted Vendor Record Report

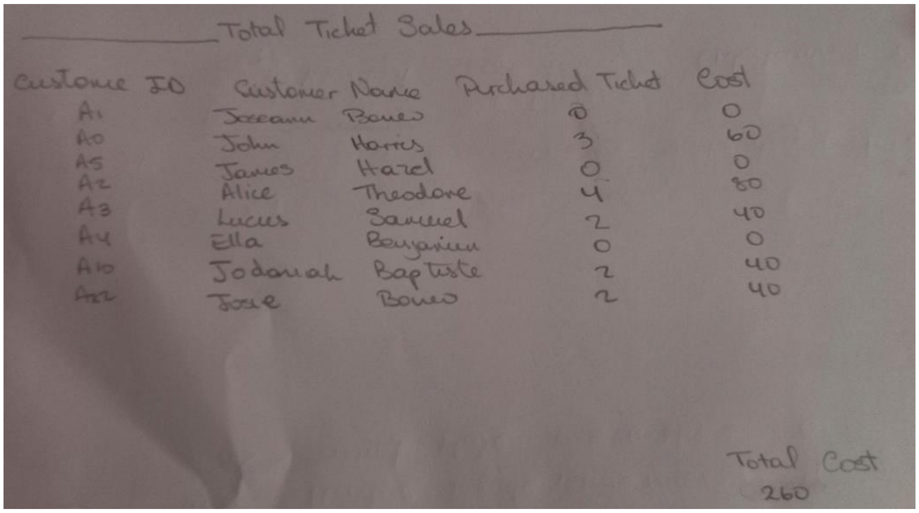


Sorted Vendor Records

Vendor ID	Vendor Name	Business Name	Stall Category	Stall ID
B1	Paris Joseph	Ice-Dreams	Food	20
B2	Saaja Khemlani	Zipe_living	Entertainment	21

Headings were used to show the individual vendor records that were sorted.

Total Ticket Sales Report



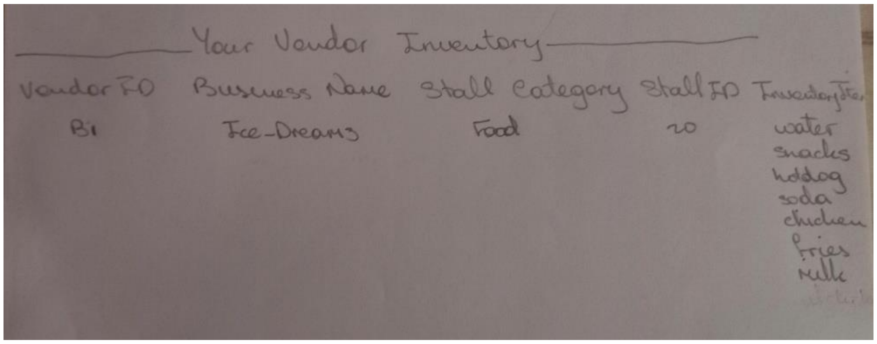
Total Ticket Sales

Custome ID	Customer Name	Purchased Ticket	Cost
A1	Joseann Bones	0	0
A0	John Harris	3	60
A5	James Hazel	0	0
A2	Alice Theodore	4	80
A3	Lucas Samuel	2	40
A4	Ella Benjamin	0	0
A10	Jodonah Baptiste	2	40
A22	Jose Bones	2	40

Total Cost
260

Headings were used to show the individual records for the total ticket sales report.

Searched Vendor Inventory Report

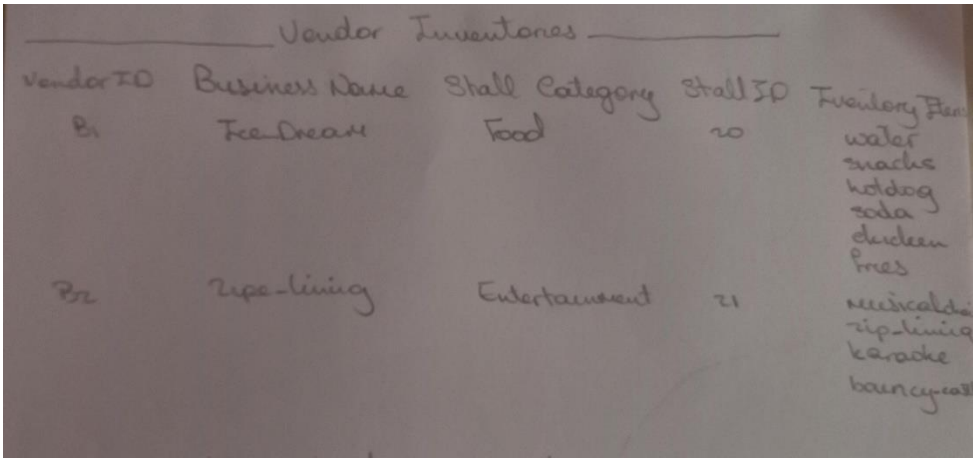


Your Vendor Inventory

Vendor ID	Business Name	Stall Category	Stall ID	Inventory Ite
B1	Ice-Dreams	Food	20	water snacks hotdog soda chicken fries milk [illegible]

Headings were used to show the individual records searched for in the vendor inventory report.

Vendor Inventory Report



Vendor Inventories

Vendor ID	Business Name	Stall Category	Stall ID	Inventory Item
B1	Ice-Dream	Food	20	water snacks hotdog soda chicken fries
B2	Zipe-lining	Entertainment	21	musicalchai zip-lining karaoke bouncy-cas

Headings were used to show the individual vendor records from the vendor inventory report.

Algorithm Design

```
Start

Begin Struct ()

char vendorID[MAX_CHARS]
char firstName[MAX_CHARS]
char lastName[MAX_CHARS]
char business_name[MAX_CHARS]
char stall_category[MAX_CHARS]
char contactNumber[MAX_CHARS]
int stallID
char vendor_password[MAX_CHARS]

End Struct

Begin Struct()Vendor_Inventory

char vendorID[MAX_CHARS]
int stallID
int num_items
struct Items {
    char itemName[MAX_CHARS]
items[DATA_CAPACITY]

End Struct

Begin Struct
```

```
struct CustomerData {
char customerID[MAX_CHARS]
char firstName[MAX_CHARS]
char lastName[MAX_CHARS]
char gender[MAX_CHARS]
char contactNumber[MAX_CHARS]
int num_tickets_bought
char customer_password[MAX_CHARS]

End Struct

struct VendorData vendor[DATA_CAPACITY]
struct CustomerData customer[DATA_CAPACITY]
struct Vendor_Inventory inventory[DATA_CAPACITY]
int mm_option
```

Function Write In File

```
function writeCustomerInfoToFile(customerInfoFile, customerRecords):
    open customerInfoFile in write mode

    if customerInfoFile is NULL:
        exit the program

    for each customerRecord in customerRecords:
        write customerRecord.customerID to customerInfoFile
        write customerRecord.firstName to customerInfoFile
        write customerRecord.lastName to customerInfoFile
        write customerRecord.gender to customerInfoFile
        write customerRecord.contactNumber to customerInfoFile
        write customerRecord.num_tickets_bought to customerInfoFile
        write customerRecord.customer_password to customerInfoFile

    write "XXX" to customerInfoFile


    while not end of file customerInfoFile:
        continue reading customerInfoFile


    close customerInfoFile


        function writeVendorInfoToFile(vendorInfoFile, vendorRecords):
    open vendorInfoFile in write mode


    if vendorInfoFile is NULL:
        print "Error"
        exit the program


    for each vendorRecord in vendorRecords:
        write vendorRecord.vendorID to vendorInfoFile
        write vendorRecord.firstName to vendorInfoFile
        write vendorRecord.lastName to vendorInfoFile
        write vendorRecord.business_name to vendorInfoFile
        write vendorRecord.stall_category to vendorInfoFile
        write vendorRecord.contactNumber to vendorInfoFile
        write vendorRecord.stallID to vendorInfoFile
        write vendorRecord.vendor_password to vendorInfoFile


    write "XXX" to vendorInfoFile
```

```
    while not end of file vendorInfoFile:
        continue reading vendorInfoFile

    close vendorInfoFile

vendorInfoFile = open file "Vendor_Information.txt"
vendorRecords = an array or collection of vendor information

writeVendorInfoToFile(vendorInfoFile, vendorRecords)




function writeInventoryInfoToFile(vendorInventFile, inventoryRecords):
    open vendorInventFile in write mode

    if vendorInventFile is NULL:
        print "Error"
        exit the program

    for each inventoryRecord in inventoryRecords:
        write inventoryRecord.vendorID to vendorInventFile
        write inventoryRecord.stallID to vendorInventFile
        write inventoryRecord.num_items to vendorInventFile

        for j from 0 to inventoryRecord.num_items:
            write inventoryRecord.items[j].itemName to vendorInventFile

        write a new line character to vendorInventFile
    write "XXX" to vendorInventFile

    while not end of file vendorInventFile:
        continue reading vendorInventFile

    close vendorInventFile

vendorInventFile = open file "Inventory_Information.txt"
inventoryRecords = an array or collection of inventory information

writeInventoryInfoToFile(vendorInventFile, inventoryRecords)


function loadDATA():
    initialize variables:
        i
        id1[4]
        id2[4]
        id3[4]

    open customerInfoFile in read mode
    if customerInfoFile is NULL:
        print "Error: Could not find Customer_Information file."
        exit the program

    read id1 from customerInfoFile

    while id1 is not equal to "XXX":
        copy id1 to customer[num_records_customers].customerID
        read firstName, lastName, gender, contactNumber, num_tickets_bought, and customer_password
        from customerInfoFile and store them in the corresponding customer record

       num_records_customers++

        read id1 from customerInfoFile

    close customerInfoFile

    open vendorInfoFile in read mode
    if vendorInfoFile is NULL:
        print "Error"
        exit the program

    read id2 from vendorInfoFile

    while id2 is not equal to "XXX":
        copy id2 to vendor[num_records_vendors].vendorID

        read firstName, lastName, business_name, stall_category, contactNumber, stallID, and
vendor_password
        from vendorInfoFile and store them in the corresponding vendor record

         num_records_vendors++

        read id2 from vendorInfoFile

    close vendorInfoFile
```

```
function loadInventoryData():
    open vendorInventFile in read mode

    if vendorInventFile is NULL:
        print "Error"
        exit the program

    read id3 from vendorInventFile

    while id3 is not equal to "XXX":
        copy id3 to inventory[num_records_inventory].vendorID

        read stallID and num_items from vendorInventFile and store them in the corresponding inventory record

        for i from 0 to inventory[num_records_inventory].num_items:
            read itemName from vendorInventFile and store it in inventory[num_records_inventory].items[i].itemName

        num_records_inventory++

        read id3 from vendorInventFile

    close vendorInventFile



function admin_entry():
    clear the console screen

    initialize code array with a size of 25
    print ("Welcome to the sign in Screen.")
    print ("Enter your Credentials")
    print ("----------------------------------")
    print ("Please enter password:")
    read code from user input

    initialize password as a constant string with the value "abc"

    if code is equal to password:
        print ("Access Granted..........")
        pause program execution
        clear console screen
        return 1
    else:
        print ("Access Denied...........")
        pause  program execution
        clear console screen
        return 0
function vendor_entry():
    initialize i
    clear console screen

    initialize code array with a size of 25
    initialize ID array with a size of 25

    print ("Welcome to the sign in Screen.")
    print ("Enter your Credentials")
    print ("----------------------------------")
    print ("Please enter user ID:")
    read ID from user input
    initialize k
    for k from 0 to length of ID:
        convert ID[k] to uppercase

    print ("Please enter password:")
    read code from user input

    for i from 0 to num_records_vendors:
        if ID is equal to vendor[i].vendorID and code is equal to vendor[i].vendor_password:
            print ("Access Granted..........")
            copy vendor[i].vendorID to vendorId
            return 1

    if i is equal to num_records_vendors:
        print ("Access Denied...........")
        return 0



function customer_entry
    initialize i
    clear  console screen

    initialize code array with a size of 25
    initialize ID array with a size of 25

    print ("Welcome to the sign in Screen.")
    print ("Enter your Credentials")
    print ("-----------------------------------")
    print ("Please enter user ID:")
    read ID

    initialize k
    for k from 0 to length of ID:
        convert ID[k] to uppercase

    print ("Please enter password:")
    read password

    for i from 0 to num_records_customers:
        print(" ID, Customer Password")

        if ID is equal to customer[i].customerID and code is equal to customer[i].customer_password:
            print ("Access Granted..........")
            copy customer[i].customerID to customerId
            return 1

    if i is equal to num_records_customers:
        print ("Access Denied...........")
        return 0



function exit_message():

    clear screen

    // Display the exit message
    print("********EXIT THE PROGRAM********")
    print("--------------------------------")
    print("THANK YOU FOR USING THE SYSTEM!")
    print("HOPE TO SEE YOU THERE ^-^")
    print("--------------------------------")

    // Write data to a file
    write_in_file()

    // Pause the program and wait for user input
    system("pause")

    // Exit the program with status code 1
    exit(1)



function add_vendor


print("--------NEW USER REGISTRATION-------")
    print ("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")
    print ("Vendor Personal Information")
    print ("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")
    print ("Vendor ID: ")
    read  vendor ID
    print ("First Name: ")
    read  first name
    print (" Last name: ")
    read  last name
    print ("Business Name: ")
    read  business name
```

```
    print ("Contact Number: ")
    read  contact nmber
    print (Password: ")
    read  vendor password

    print ("~~~~~~~~~~~~~Stall Information~~~~~~~~~~~~~")
    print ("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")
    print ("Stall Category: ")
    read  stall category
    print ("Stall ID: ")
    read  stall ID

    num_records_vendors <- num_records_vendors + 1

END FUNCTION



function add_customer():
    print ("--------NEW USER REGISTRATION--------")
    print ("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")
    print ("Customer Personal Information")
    print ("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")
    print ("Customer ID: ")
    read customer ID
    print ("First Name: ")
    read first name
    print ("Last Name: ")
    read last name
    print ("Gender: ")
```

```
    read gender
    print ("Contact Number: ")
    read contact number
    print ("Password: ")
    read customer password

    set num_tickets_bought to 0

    set customerId to customer[num_records_customers].customerID
    set num_records_customers to num_records_customers + 1

    return 1

Function edit_record(

    clear screen
    if option is equal to 2:

        for i = 0 to num_records_customers:
            if customerId is equal to customer[i].customerID:

                print ("-----ARE YOU SURE YOU WANT TO EDIT THIS RECORD-----")
                print ("-----PLEASE ENTER YOUR ANSWER-----Y - YES N - NO  Choice = ")
                read answer

                                        if ans is equal to 'y' or 'Y':
                            print ("**********EDIT CUSTOMER RECORDS**********")
                    print "-----------------------------------------"
                    print ("First Name: ")
                    read  first name
                    print ("Last Name: ")
                    read  last name
                    print ("Gender: ")
                    read  gender
                    print ("Contact Number: ")
                    read contact number

                    print ("Confirmed Editing")

                    system("pause")
                    system("cls")
                else:
                    print ("---->Route back to Your Main Menu<----")

                    system("pause")
                    system("cls")

    else if option is equal to 3:
        for i = 0 to num_records_vendors:
            if vendorId is equal to vendor[i].vendorID:
                print ("-----ARE YOU SURE YOU WANT TO EDIT THIS RECORD-----")
                print ("-----PLEASE ENTER YOUR ANSWER-----Y - YES  N - NO Choice = ")
                read answer

                if ans is equal to 'y' or 'Y':

                    print ("**********EDIT VENDOR RECORDS**********")
                    print ("--------------------------------------")
                    print "First Name: "
                    read first name
```

```
                    print ("Last Name: ")
                    read last name
                    print ("Contact Number: ")
                    read contact number
                    print ("Business Name: ")
                    read business name

                    print "(Confirmed Editing")

                    system("pause")
                    system("cls")

                else:
                    print ("---->Route back to Your Main Menu<----")
                    system("pause")
                    system("cls")
Function customer_delete:
    if option is equal to 1:
        print ("Enter Customer ID to View: ")
        read customer Id

        for k = 0 to length of customerId:
            set customerId[k] to uppercase(customerId[k])

    for i = 0 to num_records_customers:
        if customerId is equal to customer[i].customerID:
            print("\n------------------------------------------------------------------------")
            print ("Customer Information")
            print("------------------------------------------------------------------------")
            print ("Name of Customer:")
            print ("Gender: ")
from  customer[i].gender
            print ("Contact Number: ")
from  customer[i].contactNumber
            print ("Tickets Purchased: ")
from   customer[i].num_tickets_bought
            print("----------------------------------------------------------------------------")

            print ("-----ARE YOU SURE YOU WANT TO DELETE THIS RECORD-----")
            print ("-----PLEASE ENTER YOUR ANSWER-----Y - YES N - NO Choice = ")
            read answer

            if ans is equal to 'y' or 'Y':

                print ("No problem, This record shall be deleted for you.")

                if i is equal to num_records_customers - 1:
                    decrement num_records_customers by 1
                else:
                    for t = i to num_records_customers - 1:
                        swap customer[t] with customer[t+1]

                    decrement num_records_customers by 1
            else:
                print ("---->Route back to Your Main Menu<----")

function vendor_delete:
    if option equals 1 or option equals 2:

        print ("Enter Vendor ID to View: ")
        read vendoer ID

        for k from 0 to length of vendorId:
            vendorId[k] = uppercase(vendorId[k])

    for i from 0 to num_records_vendors:
        if vendorId equals vendor[i].vendorID:
            print("-------------------------------------------------------------------------")
            print ("Vendor Information")
            print("-------------------------------------------------------------------------")
            print ("Name of Vendor:")
from   vendor[i].firstName, vendor[i].lastName
            print ("Name of Business:")
from   vendor[i].business_name
            print ("Contact Number:")
from   vendor[i].contactNumber
            print ("Stall ID:")
from  vendor[i].stallID
            print ("Stall category:")
from   vendor[i].stall_category
            print("-------------------------------------------------------------------------")

            print ("-----ARE YOU SURE YOU WANT TO DELETE THIS RECORD-----")
            print ("YOUR INVENTORY WILL ALSO BE DELETED")
            print ("-----PLEASE ENTER YOUR ANSWER-----")
            print (" Y - YES  N - NO")
            print ("Choice = ")
            read choice

            if ans equals 'Y' or ans equals 'y':
                print ("No problem, This record shall be deleted for you.")

                if i equals num_records_vendors - 1:
                    num_records_vendors = num_records_vendors - 1
                else:
                    for t from i to num_records_vendors - 1:
                        temp = vendor[t]
                        vendor[t] = vendor[t+1]
                        vendor[t+1] = temp

                num_records_vendors = num_records_vendors - 1

                for s from 0 to num_records_inventory:
                    if vendorId equals inventory[s].vendorID:
                        if s equals num_records_inventory - 1:
                            num_records_inventory = num_records_inventory - 1
                        else:
                            temp1 = inventory[s]
                            inventory[s] = inventory[s+1]
                            inventory[s+1] = temp1

                    num_records_inventory = num_records_inventory - 1



FUNCTION admin_menu():
    DECLARE choice AS INTEGER

    print ("--------------------------------------------")
    print ("~~~~~~~~~WELCOME TO ADMIN MAIN MENU~~~~~~~~~~")
```

```
    print ("------------------------------------------")

    print (" 1 - View Customer Records")
    print (" 2 - View Vendor Records"   )
    print (" 3 - View Ticket Sales")
    print (" 4 - View Vendors Inventories")
    print (" 5 - Sort Customer Records")
    print (" 6 - Delete Customer Records")
    print (" 7 - Sort Vendor Records")
    print (" 8 - Delete Vendor Records")
    print (" 9 - Exit System")

    print "\n\nOption: "
     read  choice
    return choice



FUNCTION vendor_menu():
    DECLARE choice AS INTEGER

    print ("-----------------------------------------")
    print ("~~~~~~~~~~WELCOME TO VENDOR MAIN MENU~~~~~~~~~~")
    print ("------------------------------------------")

    print ("1 - Add to Your Inventory")
    print ("2 - View Your Inventories" )
    print ("3 - View Your Personal Information")
    print (" 4 - Edit Your Personal Information")
    print  ("5 - Delete Your Information")
    print (" 6 - Exit System")
```

```
    print ("Option: ")
    read choice
    return choice



FUNCTION customer_menu
    DECLARE choice AS INTEGER

    print ("------------------------------------------------")
    print ("~~~~~~~~~WELCOME TO CUSTOMERS MAIN MENU~~~~~~~~~~")
    print ("-----------------------------------------------")

    print (" 1 - View Your Personal Information")
    print (" 2 - Purchase A Ticket")
    print (" 3 - View Vendors Records")
    print (" 4 - Edit Your Personal Information")
    print (" 5 - Delete Your Information")
    print ("6 - Exit System")

    print ("Option: ")
    read option

    return choice



function customer_records

    if option equals 1:
```

```
        clear_screen()

        print ("Enter Customer ID to View: ")
       read Customer ID

        for k from 0 to length of customerId:

            customerId[k] = uppercase(customerId[k])

    for i from 0 to num_records_customers:
        if customerId equals customer[i].customerID:
                                                     print("----------------------------------------------------------------------")
print ("Customer Information")
             print("----------------------------------------------------------------------")
            print( "Name of Customer:")
get customer[i]firstNAme and customer[i]lastName
            print ("Gender:")
get customer[i].gender
            print ("Contact Number:")
get customer[i].contactNumber
            print ("Tickets Purchased:")
get customer[i].num_tickets_bought
            print("----------------------------------------------------------------------")

            pause_execution
            clear_screen



    function vendor_records:
```

```
    if option is equal to 1 or option is equal to 2:
        clear_screen()

        print "Enter Vendor ID to View: "
        read vendorId

        for k = 0 to length(vendorId) - 1:
            vendorId[k] = convert_to_uppercase(vendorId[k])

    for i = 0 to num_records_vendors - 1:
        if vendorId is equal to vendor[i].vendorID:
            print("------------------------------------------------------------------------")
            print "Vendor Information"
            print("------------------------------------------------------------------------")
            print ("Name of Vendor: " )
            print ("Name of Business: ")
            print ("Contact Number: " )
            print ("Stall ID: " )
            print ("Stall Category: " )
            print("------------------------------------------------------------------------")
            pause_execution()
            clear_screen()


function purchase_ticket():
    clear_screen()
    declare tickets as integer

    for i = 0 to num_records_customers - 1:
        if customerId is equal to customer[i].customerID:
```

```
        Print( "----------------------------------------")
        print ("TICKET")
        print  (“Christmas Bazaar")
        print ("----------------------------------------")
        print ("Venue: Holy Name Convent, Port of Spain")
        print ("Date of Event: Saturday 5th November, 2023")
        print ("Time of Event: 12pm to 6pm")
        print ("Admission Cost: $20")
        print ("----------------------------------------")
        print ("Enter the Number of Tickets: ")
        read tickets
        print ("Charges: " )
        print ("----------------------------------------")
        print ("Confirm Ticket")
        print ("----------------------------------------"
        print ("Free Secured Parking Available\n SEE YOU THERE :)")

num_tickets_bought = tickets




function ticket_sales():

    total = 0

    clear_screen

    print ("  ___________ Total Ticket Sales_____________")
    print ("Customer ID, Customer Name Purchased Tickets Cost")
    for i = 0 to num_records_customers - 1:
        print (" Customer ID, Customer First Name, Customer Last Name, Number of Tickets Bought, Entry Cost")
        total = total + (num_tickets_bought * entry_cost)


    print "Total Cost"
    pause_execution
    clear_screen


function customer_sort()
    initialize j as integer
    initialize temp as CustomerData structure


    initialize somethingSwapped as boolean
    clear screen


    for i = 0 to num_records_customers - 1
        set somethingSwapped to false
        for j = 0 to num_records_customers - 1 - i
            if string compare (customerID) > 0 then


                set temp to customer[j]
                set customer[j] to customer[j+1]
                set customer[j+1] to temp
                set somethingSwapped to true


            end if
        end for


        if somethingSwapped is false
```

```
            break
        end if
    end for

    print " __________ Sorted Customer Records___________"
    print "Customer ID, Customer Name , Gender , Contact , Purchased Tickets"

    for i = 0 to num_records_customers - 1
        print("Customer ID, Customer First Name, Customer Last Name, Gender, Contact
             Number, Number of Tickets Bought")

    end for

    pause execution
    clear screen
end function

function vendor_sort()
    j = 0
    temp = VendorData structure
    somethingSwapped = false

    clear console screen

    for i = 0 to num_records_vendors - 1 do
        somethingSwapped = false
        for j = 1 to num_records_vendors - i - 1 do
```

```
            if string compare(vendorID > 0) then
                temp = vendor[j]
                vendor[j] = vendor[j+1]
                vendor[j+1] = temp

                somethingSwapped = true
            end if
        end for

        if somethingSwapped = false then
            break
        end if
    end for

    print "__________ Sorted Vendor Records____________\n"

    print "Vendor ID, Vendor Name ,Business name , Stall Category ,Stall I"

    for i = 0 to num_records_vendors - 1 do

        print(" Vendor ID, Vendor First NAme, Vendor Last Name, Busisnes Name, Stall Category, Stall ID")

    end for

    pause system
    clear console screen
end function
```

```
void add_invent


    // Prompt user to enter vendor ID and amount of items to add
    for (i=0 while i<num_records_inventory i++)
        if (string compar of Vendor ID==0)
            print ("Enter the number of items to add to your inventory: ")
            input amount


            // Add items to inventory
            num_items = num_items + amount
            print ("Enter the name of each item: ")


            for (j=0; j<amount; j++) {
                input inventory items, inventory num_items and item name



void view_inventories()




    print ("Vendor ID, Business Name, Stall Category, Stall ID, Inventory Items")
    for(k=0 while k<num_records_vendors k++)


        for (i=0 while i<num_records_inventory i++)
            if(string compare of Vendor ID==0)




                print ("Inventory, Business Name, Stall Category, Stall ID")


                for (j=0 while j<inventory[i].num_items j++)
```

```
                    if (j == 0)
                        print ("Inventory ITems, Item Names")



function your_invent(char vendorId[])

    int j


    printf(" Your Vendor Inventor");
    printf("Vendor_ID   Business_Name   Stall_Category    Stall_ID    Inventory_Items")


    for (i=0 while i<num_records_inventory )
        i++

        if (string compare  of Vendor ID== 0

            print("Vendr Information : Inventory, Business Name, Stall Category, Stall ID")

            // Loop through all inventory items
            for (j=0; j<inventory[i].num_items)
j++
                // print item name
                if (j == 0)
                    print("Inventory Items)
```

```
void handle_Admin()

    int temp;
    int flag;

    // check if the admin password is correct
    if (admin_entry is not equal to 1) {
    print "Incorrect password... returning to root menu."
     pause
     clear screen
 else
    set flag to 1
    while (flag is equal to 1)
        temp = call admin_menu function

        switch (temp)
            case 1:
                call customer_records function with parameter 1
                break
            case 2:
                call vendor_records function with parameter 1
                break
            case 3:
                call ticket_sales function
                break
            case 4:
                call view_inventories function
                break
            case 5:
                call customer_sort function
                break
            case 6:
                call customer_delete function with parameter 1
                break
            case 7:
                call vendor_sort function
                break
            case 8:
                call vendor_delete function with parameter 1
                break
            case 9:
                call exit_message function
                set flag to 0
                break
            default:
                print "You did not enter an option that can be processed."
                print "Returning to the previous menu..."
                 pause
                 clear screen
                set flag to 0
                break




void handle_Customer()
    int user input


    print("~~~~~~~------------------------------~~~~~~~")
    print("~~~~~~~~~~~~~~Christmas Bazaar~~~~~~~~~~~~~~")
```

```
    print("~~~~~~~------------------------------~~~~~~~")
    print("Venue: Holy Name Convent, Port of Spain")
    print("Date of Event: Saturday 5th November, 2023")
    print("Time of Event: 12pm to 6pm");
    print("~~~~~~~------------------------------~~~~~~~")
    print("Select User Type")
    print(" 1 - New Customer")
    print(" 2 - Existing Customer")
    print(" 9 - Exit")
    print("Option: ")

    read user input

    int flag;

    if (user input == 9)
        exit_message
     else if (user input == 1)
        if (add_customer== 1)
            if (customer_entry== 1)
                flag = 2
            else
                print("Invalid Option Entered..............")
                system_pause
                system_clea();
                handle_Customer



else if (user input == 2)
        if (customer entry == 1)
```

```
            flag = 2;
        else
            print("Invalid Option Entered..............")
            system_pause
            system_clear
            handle_Customer

else
        print("Invalid Option Entered...............")
        system_pause
        system_clear
        handle_Customer
    }

    while (flag == 2)
        int temp = customer_menu
        switch (temp)
            case 1:
                customer_records(2)
                break
            case 2:
                purchase_ticket()
                break
            case 3:
                vendor_records(2)
                break
            case 4:
                edit_record(2)
                break
            case 5:
```

```
            customer_delete(2)
            break
        case 6:
            exit_mesage()
            break;
        default:
            print("You did not enter an option that can be processed.\n"
                  "Returning to the previous menu...");
            flag = 0




function handle_Vendor
    flag = 0
    user input = 0

    print("~~~~~~~------------------------------~~~~~~~")
    print("~~~~~~~~~~~~~~Christmas Bazaar~~~~~~~~~~~~~~")
    print("~~~~~~~------------------------------~~~~~~~")
    print("Venue: Holy Name Convent, Port of Spain")
    print("Date of Event: Saturday 5th November, 2023")
    print("Time of Event: 12pm to 6pm")
    print("~~~~~~~------------------------------~~~~~~~")
    print("Select User Type")
    print(" 1 - New Vendor")
    print(" 2 - Existing Vendor")
    print(" 9 - Exit")
    print("Option: ")

    read user input
```

```
if user input == 9:
    exit_message

if user input == 1:
    if add_vendor == 1:
        if vendor_entry == 1:
            flag = 3
        else:
            print("Invalid Option Entered...............")
            pause and clear
            handle Vendor
    else:
        print("Invalid Option Entered...............")
        pause_and_clear
        handle_Customer

else if user input == 2:
    if vendor_entry == 1:
        flag = 3
    else:
        print("Invalid Option Entered...............")
        pause and clear
        handle Vendor

else:
    print("Invalid Option Entered...............")
    pause and_clear
    handle Customer
while flag == 3:
display vendor menu and read user input
    switch temp:
        case 1:
            add_invent
            break
        case 2:
            your_invent
            break
        case 3:
            vendor records
            break
        case 4:
            edit record
            break
        case 5:
            vendor delete
            break
        case 6:
            exit message
            break
        default:
            print("You did not enter an option that can be processed.\n"
                  "Returning to the previous menu...")
            flag = 0
```

```
// Display menu options and handle user input
int mm_option
```

```
while user option is not equal to 9
    if user option== 1 then
        handle_Admin
     else if user option == 2
        handle_Customes
    else if user option == 3
        handle_Vendor

    // Display event information and menu options
    print("~~~~~~~~------------------------------~~~~~~~~")
    print("~~~~~~~~~~~~~~~~Christmas Bazaar~~~~~~~~~~~~~~~~")
    print("~~~~~~~~------------------------------~~~~~~~~")
    print("Venue: Holy Name Convent, Port of Spain")
    print("Date of Event: Saturday 5th November, 2023")
    print("Time of Event: 12pm to 6pm")
    print("~~~~~~~~------------------------------~~~~~~~~")
    print ("Select User Type")
    print(" 1 - Admin")
    print(" 2 - Customer")
    print(" 3 - Vendor")
    print(" 9 - Exit")
    print("Option: ")
    read option


// Handle exit message
if option choice is 9
  display exit message

return 0
```

Data Structure Design

Data Structure

Data structures allow us to process data about an entity which consists of items. Structs are useful as they can group variables of different types representing information related to a single object representing fields under a single tag.

Incorporating the use of data structures into your program comes with several of its own advantages. They are as follows:

- Heterogeneous storage of data- A defined number of struct variables allow pertinent fields of information about a collection of items to be stored. This is the most beneficial way to cache separate records of an entity, where the values assigned to the fields will be different for each.
- Maintainability of code- Structures are the most practical method of representing complex records by using a single tag, which can be referred to when the values in the fields are to be used in the code.
- Ease of Code Readability- Structure code greatly mitigates the convolution of code and is more user friendly to those reviewing the algorithm. It reduces the complexity of the code as the same label is used throughout.

Three structs were used in our program, one being the struct VendorData, the use of which will be discussed.

The purpose of this struct is to hold individual records of information relating to each vendor at the bazaar.

The main components that make up a struct are the keyword, the structure tag, structure members/fields and in some cases, one or more structure variables.

The keyword ‘struct’ is first used to declare the structure. This is then followed by the structure tag which is, in this instance, 'VendorData’. Structure members are then defined and enclosed within curly brackets ‘{}’. The members of the VendorData struct include vendorID, firstName, lastName, business_name, stall_category, contactNumber, stallID and vendor_password.

The struct variables used to establish the number of records of vendors is an array variable, vendor[DATA_CAPACITY].vendorID. This declares the number of records will be value assigned to the global variable

To access data within a struct, a dot (.) operator is used. For example, vendor[v].vendorID. This command will access the vendor ID of the vendor stored in location v within the vendor record.

Files

A file refers to any stand-alone information that is available to the operating system and individual programs for the purpose of reading (inputting) and writing (outputting) data. Additionally, they also allow data to be updated and stored for future use. Data files were a key component apropos of the data entry and output of the program. All data contained within the file can be easily accessed by using a few basic C commands. There are several advantages to using files when programming. These include:

- Transferability- Data stored in files can easily be distributed as a single unit between computers for viewing, retrieval and modification without any changes.
- Data Preservation- The data printed in a file will still be available even after the program has terminated.
- Enhanced Data Entry Speed-Large amount of data from a file can be entered at a faster rate by using the fopen function with an associated pointer in C programming, as opposed to manually entering each individual line of data.

In our program, three main input and output files were used. Now, we will discuss the use of one of those files, ‘Customer_Information.txt’. The data file ‘Customer_Information.txt’ contains 11 rows of data which gives the necessary information regarding each registered customer for the bazaar. As seen in the ‘loadData’ function, the data from the file is being read with a while loop, for the purpose of loading the Customer structure with information. To read information from a file, the C function ‘fopen’ must be used to open the file to access the data. This is then followed by the name of the file ‘Customer_Information.txt’, and the mode in which you open this file, which in this case was ‘r’, both enclosed in parentheses. Therefore, the function fopen(‘Customer_Information’txt, ’r’), would simply instruct the computer to open the Customer_Information.txt file for the purpose of reading (inputting) data.

The function ‘write_in_file utilized the ‘w’ mode when opening the ‘Customer_Information.txt’ file for writing (outputting) data from the data structures used in the program. The function fopen(‘Customer_Information.txt’, ‘w’), simply instructed the computer to carry on this task applied to that file.

This ultimately decreases the overall time spent coding as the data within the file can be accessed faster compared to entering each individual line of data yourself. In reference to the function ‘add_customer’, data can also be edited/updated with new information and stored separately from the code itself. This ensures that the data file is not lost even as the program terminates.

Array

An array is referred to as a type of data structure that stores a collection of elements belonging to the same data type in adjacent memory location within the computer system in a sequential manner. These elements can then be accessed using a unique index and identifier. As stated, they allow different types of data about a number of objects to be stored under one name for convenient handling, thus reducing the complexity and increasing the productivity, readability and maintainability of the program. There are several advantages to using arrays when programming. These include:

- Memory Allocation- Arrays prevent memory wastage as elements are stored in a sequential manner in memory locations
- Functionality- Arrays can be used for processing purposes in many algorithms (e.g. sorting, searching, deleting and reversing elements).
- Code Optimization- An array allows large number of values to be stored and accessed by writing a small block of code as opposed to declaring each variable individually.

Arrays of structs, struct VendorData, Vendor_Inventory and CustomerData were defined in the program to hold several records of information relating to the registered customer and vendors of the Holy Name Convent Bazaar. In this case of struct Vendor_Inventory, the array was declared as inventory[DATA_CAPACITY]. As an example, the array of struct was referred to as inventory[i].vendorID in the function 'your_invent' for searching.

The array consists of several components. The basic components are the name inventory and the array type, which is struct as previously stated. The size of the array enclosed as square brackets '[ ]' is 100, specifying that up to 100 vendors' inventory are stored. The subscript is the variable i, where the record number is being referred to.

In the function, 'your_invent' is called with a parameter, string characters vendorID which represents the unique identification of the vendor whose inventory record the user wants to view. Then a bounded loop is executed to increment the subscript 'i' and searches each record the array currently refers to until the record matches this specific criterion. Their inventory details are printed.

Coding and Testing

Test Plan

- Objectives- The objective of this test plan is to ensure that the bazaar management program is functioning as expected allowing admin, customers and vendors to navigate the system to view and modify data while meeting the previously stated non-functional requirements.

- Scope- This test plan covers testing of the following management operations:

  Menu navigation

  Log in operation

  Viewing records

  Adding new records

  Sorting

  Searching

  Deleting

  Editing

- Test Strategy- The testing will be performed manually by entering the desired menu option and the required customer or vendor ID to verify the result. The results will be verified by comparing it with the intended report designs and referencing the input files into the system used to ensure the correct data is displayed.

- Test Environment- The bazaar management program will be compiled and tested using the Integrated Development Environment (IDE) known as Code Blocks version 20 on a computer running Windows 10 with a minimum of 4GB of RAM.

- Test Cases-

Normal dataset entry

Extreme dataset entry

Erroneous dataset entry

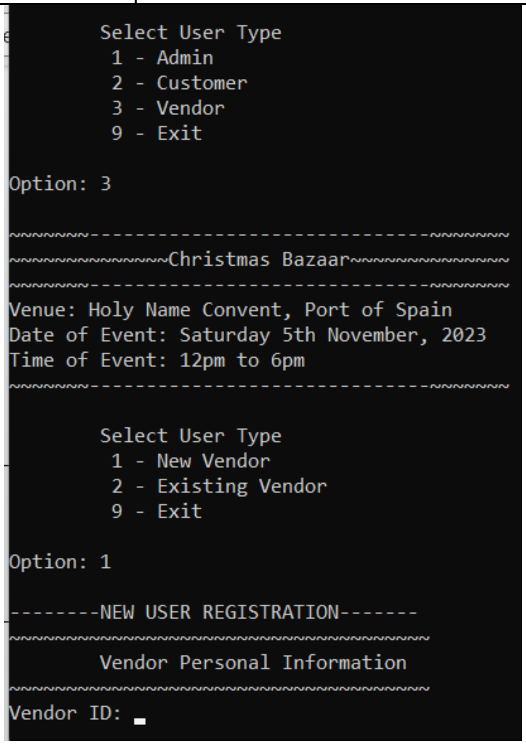
- Test Execution Schedule- Testing will be performed each time a function is completed to ensure consistency. However, the final test while occur on April 30th from 5pm to 7pm, where all the completed functions and modules will be executed as one source code and be evaluated.

- Risks and Mitigation- There is a risk that the management program may crash or produce incorrect results due to software bugs and runtime errors. To mitigate this risk, the program will be thoroughly tested and any issues due to bugs and errors will be reported and fixed. A back-up computing device with the same testing environment will be available if necessary.

- Test Deliverables- The test results will be documented in a test report table, which will include the scenarios of all the test cases. The test results will be accompanied with screenshots of executed operation and would include any issues found during the testing

Testing

TYPE OF INPUT	DESCRIPTION	EXPECTED RESULTS	ACTUAL RESULT	SUCCESS?
Normal	A user wants to be added to the program as a vendor	After opening the program, the user would be directed to the main menu. Following this, when the appropriate user type is selected 'Vendor', a following menu should	Select User Type 1 - New Vendor 2 - Existing Vendor 9 - Exit Option: 1 NEW USER REGISTRATION Vendor Personal Information Vendor ID:	Test is successful

<table>
<tr><td></td><td></td><td>be shown to create a new user.</td><td></td><td></td></tr>
<tr><td colspan="5">
<pre>
        Select User Type
         1 - Admin
         2 - Customer
         3 - Vendor
         9 - Exit

Option: 3

~~~~~~~------------------------------~~~~~~~
~~~~~~~~~~~~~~Christmas Bazaar~~~~~~~~~~~~~~
~~~~~~~------------------------------~~~~~~~
Venue: Holy Name Convent, Port of Spain
Date of Event: Saturday 5th November, 2023
Time of Event: 12pm to 6pm
~~~~~~~------------------------------~~~~~~~

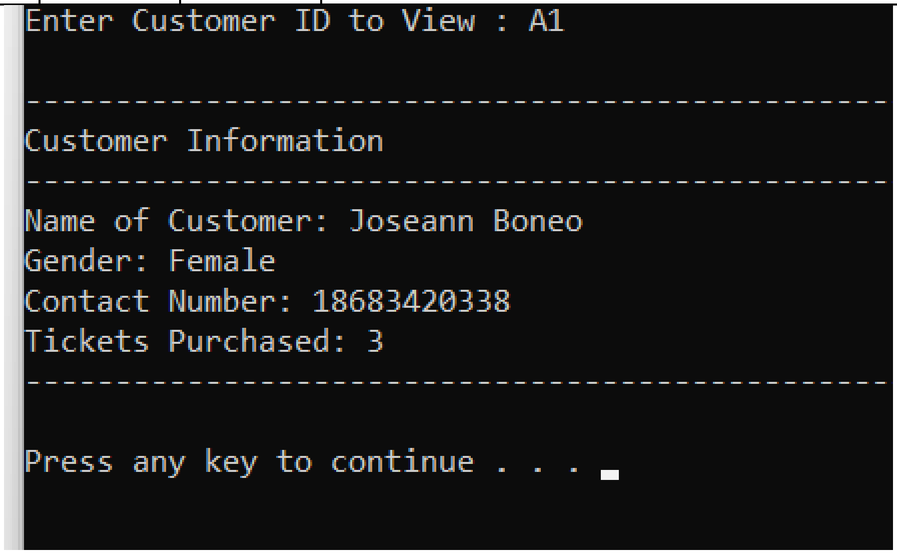
        Select User Type
         1 - New Vendor
         2 - Existing Vendor
         9 - Exit

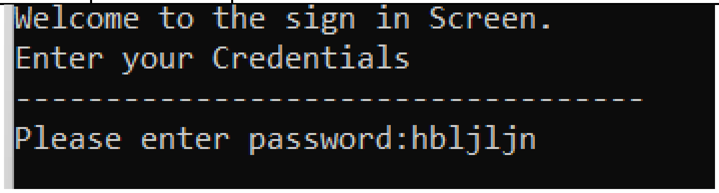
Option: 1

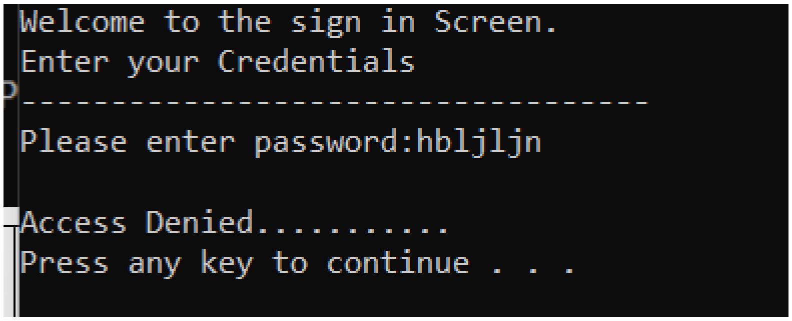
--------NEW USER REGISTRATION-------
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        Vendor Personal Information
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Vendor ID: _
</pre></td></tr>
<tr><td></td><td>Administration should be able to search the customers personal file</td><td>Entre Customer ID:
Customer Information:</td><td>Enter Customer ID to View : A1

Customer Information

Name of Customer: Joseann Boneo
Gender: Female</td><td>Test is successful</td></tr>
</table>

Extreme		Name: Gender: Contact #: Tickets:	Contact Number: 18683420338 Tickets Purchased: 3 --- Press any key to continue . . .	
	 Enter Customer ID to View : A1 -- Customer Information -- Name of Customer: Joseann Boneo Gender: Female Contact Number: 18683420338 Tickets Purchased: 3 -- Press any key to continue . . .			
	Admin entered the wrong login information	Access Denied.. .... Press any key to	Welcome to the sign in Screen. Enter your Credentials ---------------------------------- Please enter password:	Test successful

<table>
<tr><td><b>Erroneous</b></td><td></td><td>continue . . .</td><td>Access Denied..........
Press any key to continue . . .</td><td></td></tr>
<tr><td colspan="5">
Welcome to the sign in Screen.
Enter your Credentials

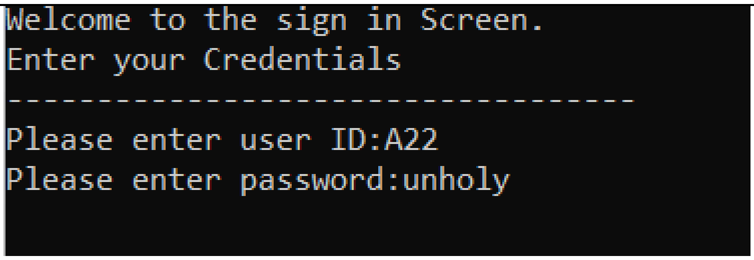
Please enter password:hbljljn

Welcome to the sign in Screen.
Enter your Credentials

Please enter password:hbljljn

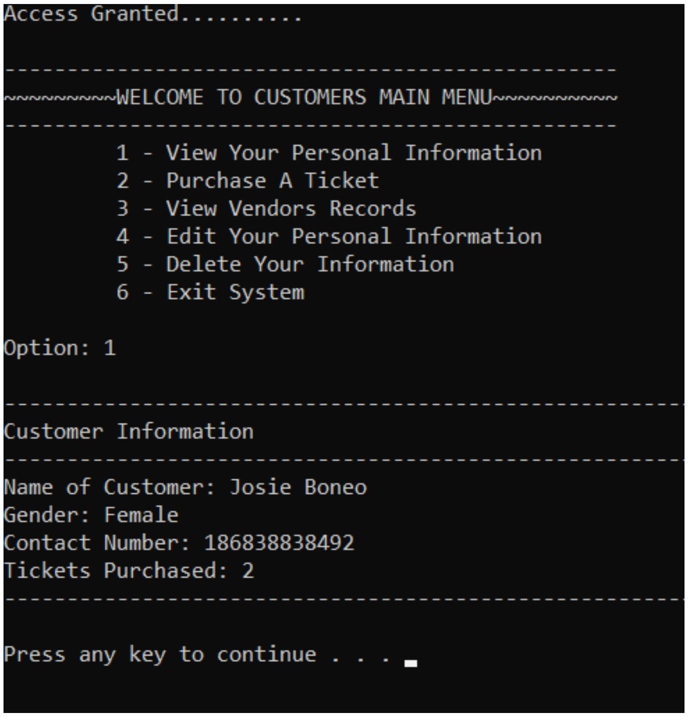
Access Denied...........
Press any key to continue . . .</td></tr>
<tr><td><b>Normal</b></td><td>Customer should be able to view their personal information</td><td>Please enter user ID:
Please enter password:
…....
Welcome to Customers Main Menu-</td><td>Access Granted..........

~~~~~~~~~WELCOME TO CUSTOMERS MAIN MENU~~~~~~~~~~
----
1 - View Your Personal Information
2 - Purchase A Ticket
3 - View Vendors Records</td><td>Test is successful</td></tr>
</table>

			4 - Edit Your Personal Information 5 - Delete Your Information 6 - Exit System Option: 1 -- Customer Information -- Name of Customer: Josie Boneo Gender: Female Contact Number: 186838838492 Tickets Purchased: 2 -- Press any key to continue . . .	



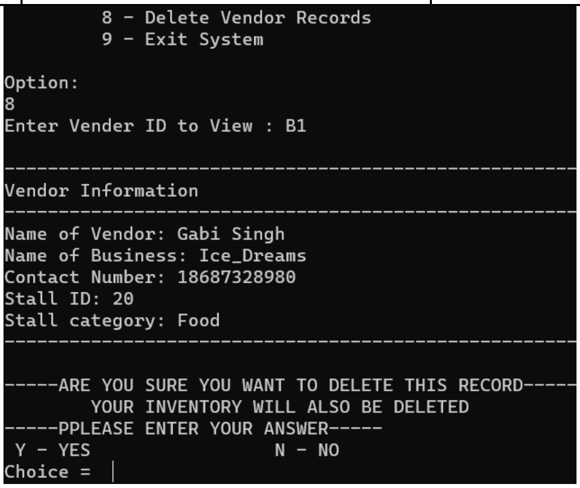
```
Welcome to the sign in Screen.
Enter your Credentials
-----------------------------------
Please enter user ID:A22
Please enter password:unholy
```



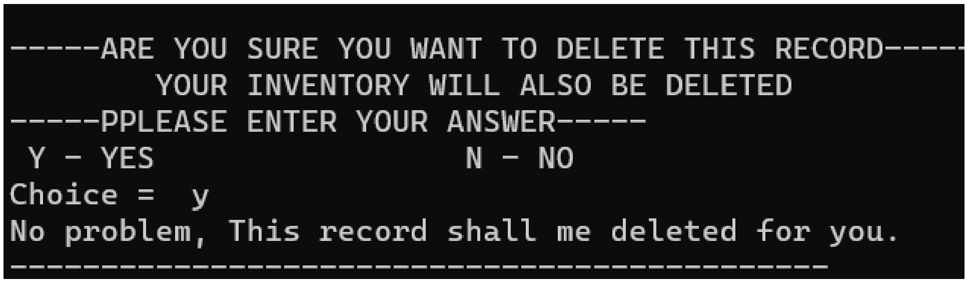
```
Access Granted..........

-------------------------------------------------
~~~~~~~~~WELCOME TO CUSTOMERS MAIN MENU~~~~~~~~~~
-------------------------------------------------
         1 - View Your Personal Information
         2 - Purchase A Ticket
         3 - View Vendors Records
         4 - Edit Your Personal Information
         5 - Delete Your Information
         6 - Exit System

Option: 1


------------------------------------------------------
Customer Information
------------------------------------------------------
Name of Customer: Josie Boneo
Gender: Female
Contact Number: 186838838492
Tickets Purchased: 2
------------------------------------------------------

Press any key to continue . . .
```

	Administration should be able to delete vendors from the system	ARE YOU SURE YOU WANT TO DELETE THIS RECORD YOUR INVENTORY WILL ALSO BE DELETED	-----ARE YOU SURE YOU WANT TO DELETE THIS RECORD----- YOUR INVENTORY WILL ALSO BE DELETED -----PPLEASE ENTER YOUR ANSWER-----	Test successful

Extreme		PPLEASE ENTER YOUR ANSWER Y - YES N - NO Choice =…..	Y - YES N - NO Choice = y No problem, This record shall me deleted for you.	

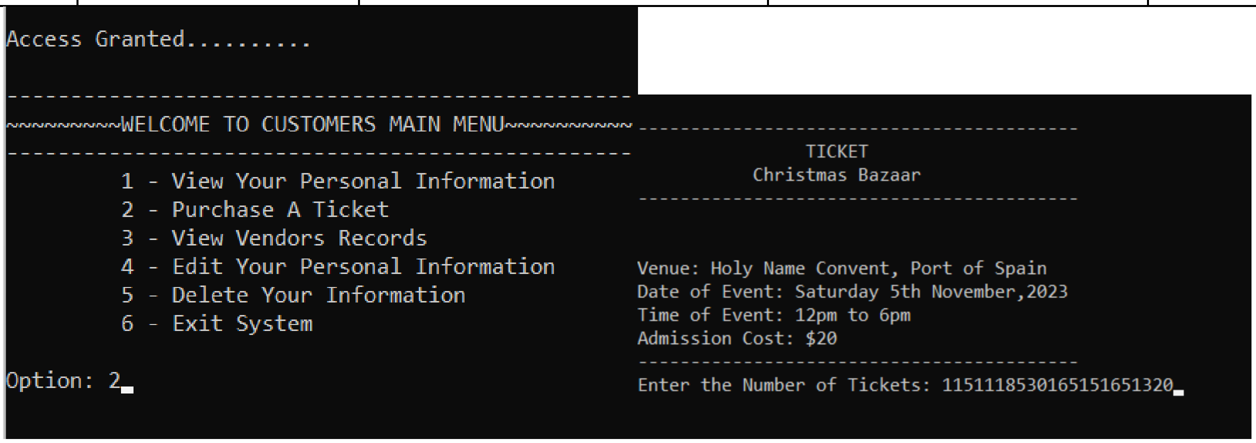


```
        8 - Delete Vendor Records
        9 - Exit System

Option:
8
Enter Vender ID to View : B1

-----------------------------------------------------
Vendor Information
-----------------------------------------------------
Name of Vendor: Gabi Singh
Name of Business: Ice_Dreams
Contact Number: 18687328980
Stall ID: 20
Stall category: Food
-----------------------------------------------------

-----ARE YOU SURE YOU WANT TO DELETE THIS RECORD-----
        YOUR INVENTORY WILL ALSO BE DELETED
-----PPLEASE ENTER YOUR ANSWER-----
 Y - YES                  N - NO
Choice =  |
```



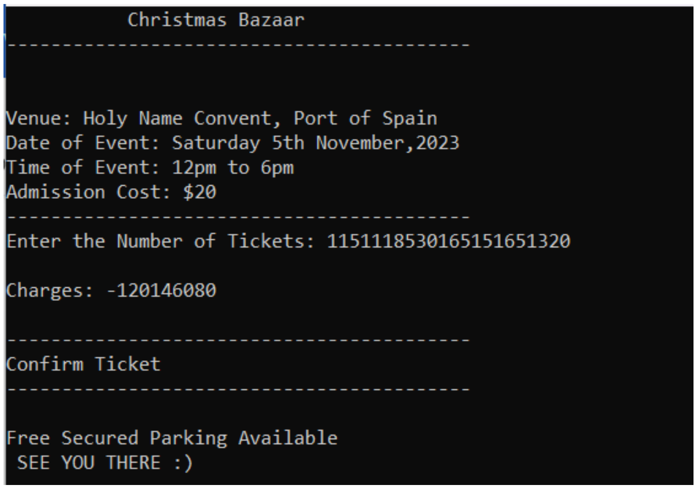
```
-----ARE YOU SURE YOU WANT TO DELETE THIS RECORD-----
        YOUR INVENTORY WILL ALSO BE DELETED
-----PPLEASE ENTER YOUR ANSWER-----
 Y - YES                 N - NO
Choice =  y
No problem, This record shall me deleted for you.
---------------------------------------------
```

Erroneous	A customer Can't Purchase a large number of tickets.	Enter the number of Tickets: ….... Sale Denied …...	Enter The Number of Tickets: …... Charges: ….. Confirm Ticket	Fail



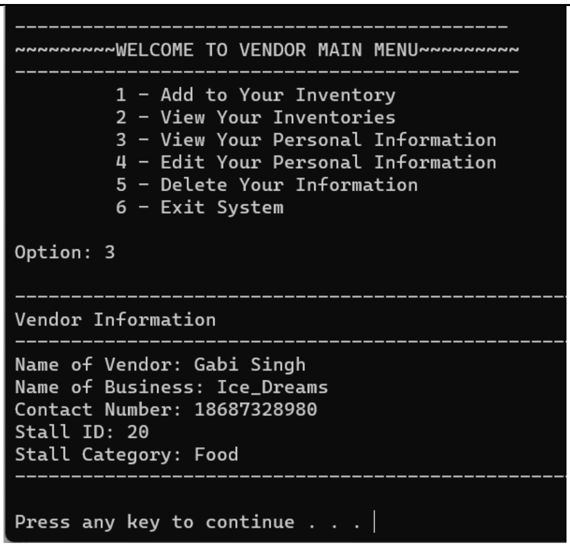
```
Access Granted..........

-------------------------------------------------
~~~~~~~~~WELCOME TO CUSTOMERS MAIN MENU~~~~~~~~~~
-------------------------------------------------
         1 - View Your Personal Information
         2 - Purchase A Ticket
         3 - View Vendors Records
         4 - Edit Your Personal Information
         5 - Delete Your Information
         6 - Exit System

Option: 2
```

```
----------------------------------------
                TICKET
           Christmas Bazaar
----------------------------------------


Venue: Holy Name Convent, Port of Spain
Date of Event: Saturday 5th November,2023
Time of Event: 12pm to 6pm
Admission Cost: $20
----------------------------------------
Enter the Number of Tickets: 1151118530165151651320
```



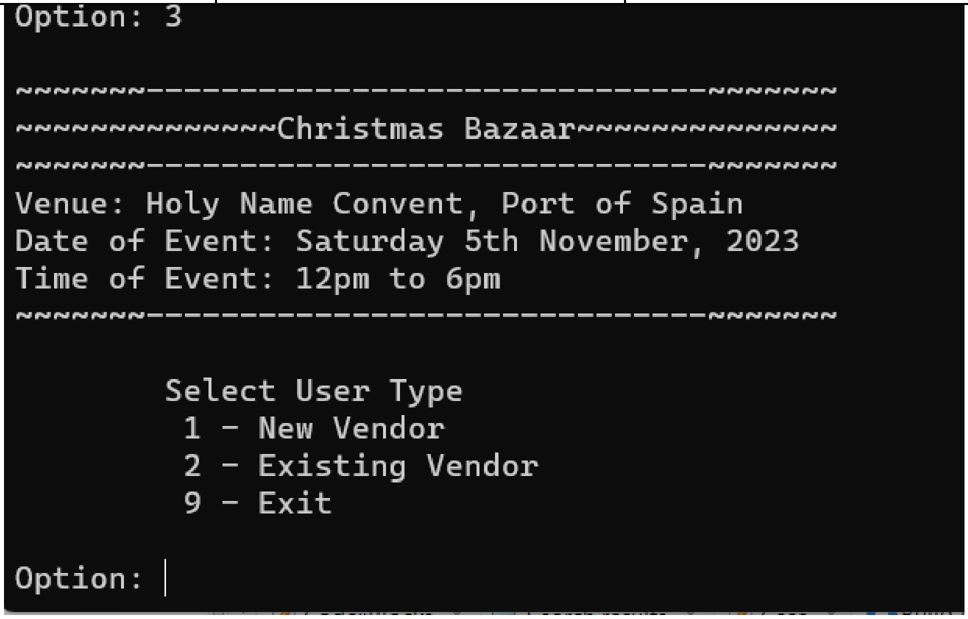
```
           Christmas Bazaar
----------------------------------------


Venue: Holy Name Convent, Port of Spain
Date of Event: Saturday 5th November,2023
Time of Event: 12pm to 6pm
Admission Cost: $20
----------------------------------------
Enter the Number of Tickets: 1151118530165151651320

Charges: -120146080

----------------------------------------
Confirm Ticket
----------------------------------------

Free Secured Parking Available
 SEE YOU THERE :)
```

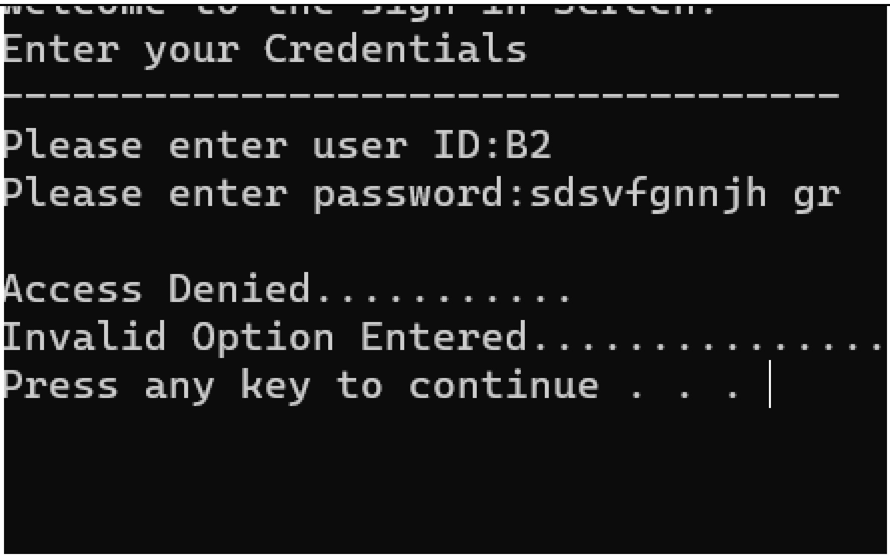
Normal	Program should allow a vendor to Login to their personal information	Vendor Information Name: Name of Business: Contact#: Stall: Stall Category:	Option: 3 -- Vendor Information -- Name of Vendor: ….. Name of Business: Ice_Dreams Contact Number: 18687328980 Stall ID: 20 Stall Category: Food -- Press any key to continue . . .	Test is successful



```
--------------------------------------------
~~~~~~~~~WELCOME TO VENDOR MAIN MENU~~~~~~~~~
---------------------------------------------
         1 - Add to Your Inventory
         2 - View Your Inventories
         3 - View Your Personal Information
         4 - Edit Your Personal Information
         5 - Delete Your Information
         6 - Exit System

Option: 3

--------------------------------------------------
Vendor Information
--------------------------------------------------
Name of Vendor: Gabi Singh
Name of Business: Ice_Dreams
Contact Number: 18687328980
Stall ID: 20
Stall Category: Food
--------------------------------------------------

Press any key to continue . . .
```

Extreme	Vendor should be able to view customer A5 James Hazel information	Enter Customers ID: …. Name of Customer: James Hazel Gender: ….. Contact Number: ….. Tickets Purchased: Press any key to continue . . .	Select User Type 1 - New Vendor 2 - Existing Vendor 9 - Exit Option:	Failed

	 Option: 3 ~~~~~~~------------------------------~~~~~~~ ~~~~~~~~~~~~~~Christmas Bazaar~~~~~~~~~~~~~~ ~~~~~~~------------------------------~~~~~~~ Venue: Holy Name Convent, Port of Spain Date of Event: Saturday 5th November, 2023 Time of Event: 12pm to 6pm ~~~~~~~------------------------------~~~~~~~ Select User Type 1 - New Vendor 2 - Existing Vendor 9 - Exit Option:			
Erroneous	The ability to enter a Longin string for vendor	Enter your Credentials Please enter user ID: Please enter password:sdsvfgnnjh gr	Please enter user ID:B2 Please enter password:sdsvfgnnjh gr Access Denied........... Invalid Option Entered............... Press any key to continue . . .	Fail



```
Welcome to the Sign in Screen.
Enter your Credentials
-----------------------------------
Please enter user ID:B2
Please enter password:sdsvfgnnjh gr

Access Denied...........
Invalid Option Entered..............
Press any key to continue . . .
```