

## Table of Contents

1. Abstract.....	3
2. Problem Analysis	
3. Introduction.....	3
4. System Architecture.....	3
5. System Implementation.....	4
6. System Testing.....	4
7. Ethical Considerations.....	4
8. Tasks to be performed.....	4
9. Methodology.....	4
10. Results.....	5
11. Conclusion.....	9
12. References.....	9

## **Abstract**

The Chat-Enabled Shopping System gives users that are customers with an easy to use system and multiple server-client chat features to enhance their shopping experience. This project limits the manual and physical interaction, providing an efficient and convenient shopping experience. The system implements the TCP protocol for admin-customer connections, the FTP protocol for sharing billing and other related files and the SMTP protocol has been used for the communication of customer queries and feedback through email.

## **Problem Analysis**

Conventional online shopping systems lack efficient communication with vendors or customer support staff, which is not a pleasant experience for customers and the system users. This inevitably results in dissatisfaction and decreased customer loyalty. In order to provide a solution for the problems listed above, we created a system that eliminates all these loopholes, and is user friendly and time saving. The aim of the Chat-Enabled Shopping System project is to work around these issues and provide a more convenient and personalized shopping experience to customers. By incorporating a real-time chat feature, customers will be able to interact with the system and receive personalized assistance during the shopping process. The project also aims to facilitate easy file sharing between the server and clients using FTP protocol and allow customers to provide feedback through email using the SMTP protocol. The main challenge is to design and implement a robust chat system that seamlessly integrates with the existing online shopping platform and enhances the overall customer experience.

## **Introduction**

Our Chat-Enables Shopping System is both a GUI and console based system developed in Python using Visual Studio Code. This strives to provide a stress-free online shopping experience for clients. The system will offer personalized server-client chat features to enhance customer experience and improve customer satisfaction.

With this system, customers will be able to connect to our server using TCP protocol and will interact with it however they like. The user can select items to buy, add items in the cart, remove any items and make transactions. Furthermore, any type of file sharing, such as billing or shopping summary, will be provided using the FTP protocol from the server to the clients.

The system's efficiency can be improved using customer feedback and queries, as they will be allowed to email using the SMTP protocol. All these features will ensure a complete shopping system using the chat messenger, which will enhance the overall customer experience. This report aims to discuss the design, implementation, and testing of the Chat-Enabled Shopping System.

## **System Architecture**

The Chat-Enabled Shopping System is a multiple client-server architecture that uses TCP protocol to connect customers with vendors or customer support. The system is made up of a front-end client and a back-end server. The front-end client provides the console and GUI based user interface for customers to interact with the system, while the back-end server manages the processing of requests and communication between the users and the system as well as implementing protocols such as the FTP protocol for sharing files like billing or shopping summary and SMTP protocol for receiving customer feedback through email.

## **System Implementation**

The implementation of the Chat-Enabled Shopping System is based on Python and the Tkinter GUI toolkit, utilizing socket programming, FTP and SMTP protocols. The system comprises a server and multiple clients that can communicate in real-time over the TCP protocol. The chat server will use Socket.IO for real-time communication. SMTP for mail transfer, and FTP protocol for file transfer.

## **System Testing**

We will conduct functional, usability, and performance testing to ensure the Chat-Enabled Shopping System meets the requirements and provides a satisfactory user experience. We will use automated testing tools to conduct functional and performance testing, and human testers to conduct usability testing. To handle any unwanted inputs or data we have included the correct exception handling for that particular problem so that the program doesn't hang in between.

## **Ethical Considerations**

As with any technological system, there are several ethical considerations that must be taken into account while developing and implementing the Chat-Enabled Shopping System.

- User privacy
- Transparency
- Fairness
- Informed consent
- Responsible use of user data
- Accessibility
- Ethical behavior of system administrators

By taking these ethical considerations into account, the Chat-Enabled Shopping System can ensure that it provides a safe, secure, and fair shopping experience for all its use and respectful of user privacy and autonomy.

## **Tasks to be performed by the system**

### **Server:**

The server program is responsible for handling incoming connections from clients, processing user requests, and responding to client queries. The server uses a multithreaded approach to handle multiple client connections simultaneously. Each client connection is managed by a separate thread, allowing the server to process multiple requests concurrently. We have implemented the FTP server as well, where it deploys the billing and order detail that the client can download. There is also an online SMTP server portal that handles the user feedback mails.

### **Client:**

The clients have access to a user-friendly console and GUI that allows them to interact with the system in a convenient and intuitive manner. The GUI includes product images that the client can shop from. It has different categories and the client can also select the quantity of these products. The clients can initiate a chat with a seller or customer service representative at any point during the shopping process. Once the customer initiates a chat, the system will connect with them using TCP protocol and display menu options, where the client can choose from. The customer will then be able to ask questions, clarify doubts, or seek guidance related to the product or the online shopping process. The seller or customer service representative will respond to the queries in real-time, providing personalized assistance to the customer.

## **Methodology**

### **Chatting Protocol:**

The Chat-Enabled Shopping System uses TCP protocol for connecting customers with sellers or customer service representatives. The system allows customers to initiate a chat with a seller or customer service representative at any point during the shopping process. Once the customer initiates a chat, the system will connect them with a seller or customer service representative in real-time. The customer can then ask questions, clarify doubts, or seek guidance related to the product or the online shopping process. The seller or customer service representative will respond to the queries in real-time, providing personalized assistance to the customer.

### **File Sharing Protocol (including authentication):**

The Chat-Enabled Shopping System allows for easy sharing of files like billing or shopping summary using FTP protocol. The server stores all the necessary files and shares them with the client as required. This feature ensures that the customer has access to all the relevant information related to their purchase.

### **Email Feedback Protocol:**

The Chat-Enabled Shopping System allows customers to provide feedback using the SMTP protocol. Customers can email their feedback to the system, which is then processed and

analyzed by the system. This feature allows the system to improve its services based on customer feedback and ensure customer satisfaction.

### **Libraries:**

#### **Server:**

```
import socket
import threading
from tkinter import *
from PIL import ImageTk, image
from items import *
import os
import json
from twisted.cred.checkers import AllowAnonymousAccess,
InMemoryUsernamePasswordDatabaseDontUse
from twisted.protocols.ftp import FTPFactory, FTPRealm
```

#### **Client:**

```
import socket
from ftpClient import ftp_client
from smtpClient import Feedback
```

### **Tools:**

- Visual Studio Code
- Python Programming
- MailTrap

### **Functions:**

#### **Server:**

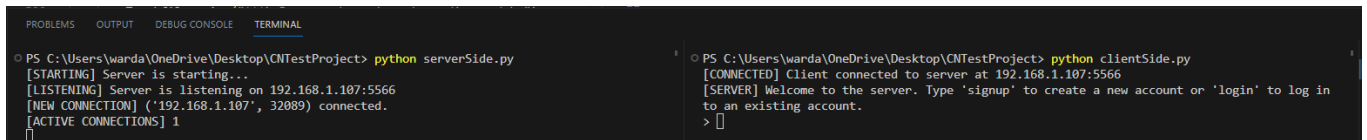
```
def handle_client(conn, addr)
def switch1(item,conn, addr)
def switch2(item,conn,username)
def clothingGui(conn,username)
def groceryGui(conn,username)
def electronicsGui(conn,username)
def stationaryGui(conn,username)
def Billing(conn,username)
```

#### **Client:**

```
def ftp_client(hostname, username, password)
def Feedback(sender_email, sender_password)
```

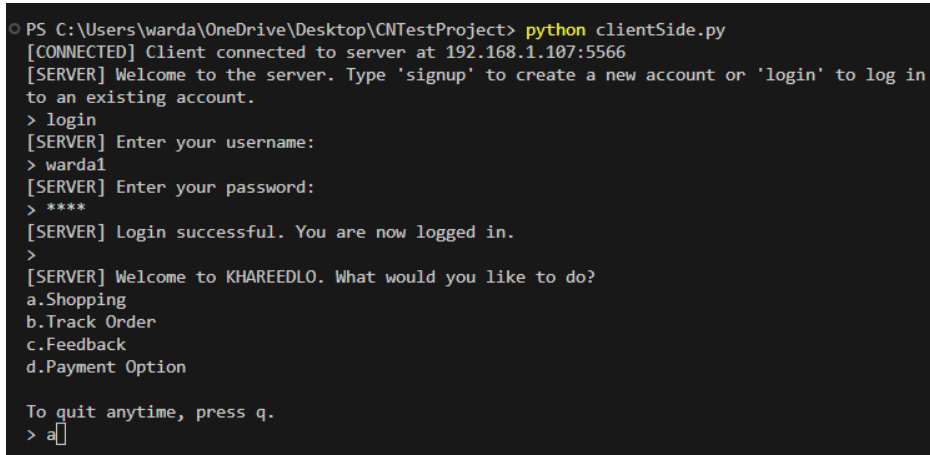
## Screenshots:

The connection is set up after running the server.



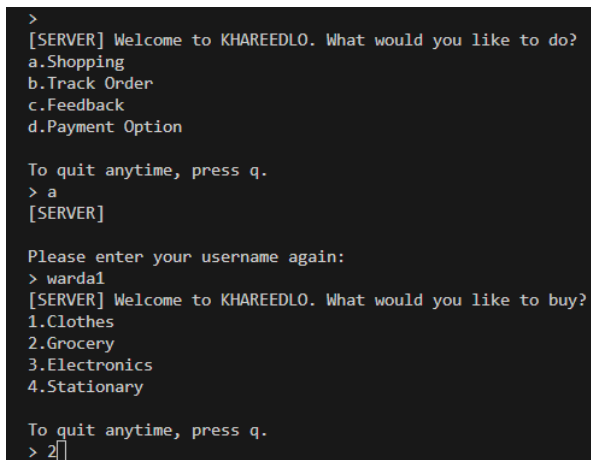
The screenshot shows two terminal windows side-by-side. The left window is titled 'TERMINAL' and shows the execution of 'python serverSide.py'. The output includes: '[STARTING] Server is starting...', '[LISTENING] Server is listening on 192.168.1.107:5566', '[NEW CONNECTION] ('192.168.1.107', 32089) connected.', and '[ACTIVE CONNECTIONS] 1'. The right window shows the execution of 'python clientSide.py'. The output includes: '[CONNECTED] Client connected to server at 192.168.1.107:5566' and '[SERVER] Welcome to the server. Type 'signup' to create a new account or 'login' to log in to an existing account.'

Initially the client is asked to sign up if they are new to the site else they can login. A file for every new client is also created and all their actions are stored there.



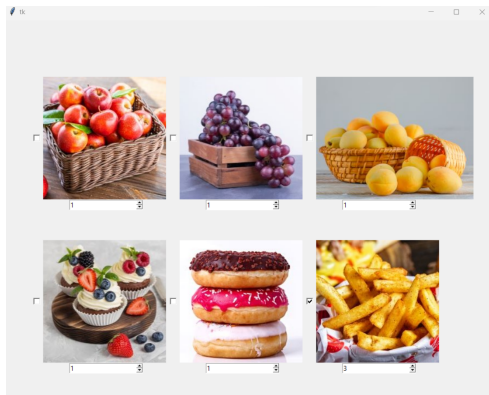
The screenshot shows a terminal window with the command 'python clientSide.py'. The output shows the client connecting to the server at 192.168.1.107:5566. The server prompts the user to type 'signup' or 'login'. The user enters 'login'. The server prompts for a username, and the user enters 'wardal'. The server prompts for a password, and the user enters '\*\*\*\*'. The server responds with 'Login successful. You are now logged in.' The server then prompts 'Welcome to KHAREEDLO. What would you like to do?' and lists options: 'a.Shopping', 'b.Track Order', 'c.Feedback', and 'd.Payment Option'. The user enters 'a'.

Now the client can select any options from the following. Here we have selected shopping. This displays another menu to the client where they can select the category they want to shop from. In this example we have selected Grocery here, this will display a GUI setup in front of the client where it can select anything from.

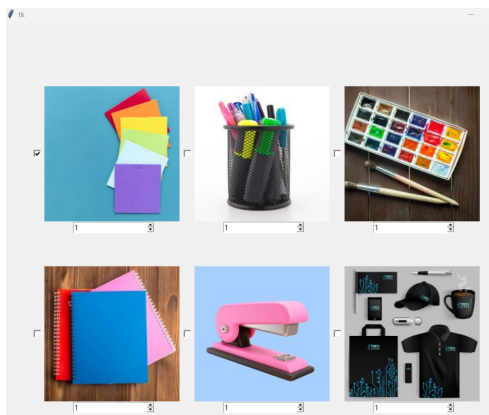
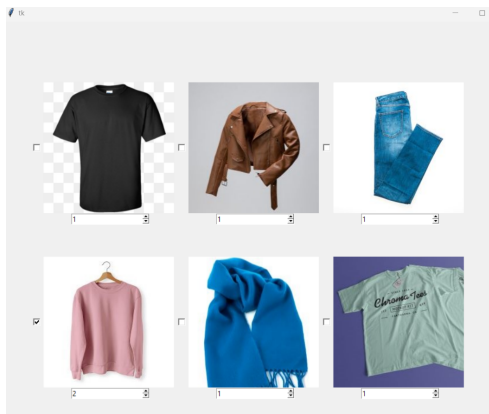
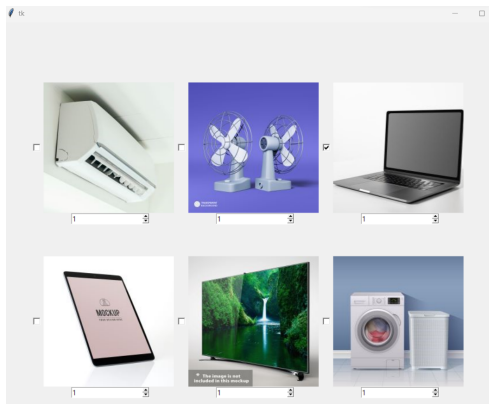


The screenshot shows a terminal window with the command 'python clientSide.py'. The output shows the client connecting to the server at 192.168.1.107:5566. The server prompts the user to type 'signup' or 'login'. The user enters 'login'. The server prompts for a username, and the user enters 'wardal'. The server prompts for a password, and the user enters '\*\*\*\*'. The server responds with 'Login successful. You are now logged in.' The server then prompts 'Welcome to KHAREEDLO. What would you like to do?' and lists options: 'a.Shopping', 'b.Track Order', 'c.Feedback', and 'd.Payment Option'. The user enters 'a'. The server then prompts 'Please enter your username again:' and the user enters 'wardal'. The server then prompts 'Welcome to KHAREEDLO. What would you like to buy?' and lists options: '1.Clothes', '2.Grocery', '3.Electronics', and '4.Stationary'. The user enters '2'.

This is the grocery GUI where the customer can select product and quantity.



The customer can also select from other categories like electronics, stationary and clothing:



Let's proceed towards the bill. Here the server will provide the user credentials to the client, so it can connect with the ftp server and download the order and bill details.

```
5.Proceed to Billing
> 5
[SERVER]
To check your Order or Bill Details, here are your credentials.
    Please donot share this with anyone.

Username: warda1
Password: 1234warda1
>Enter your KAREEDLO username: warda1
>Enter your KAREEDLO password: 1234warda1
Connected to 127.0.0.1 as warda1
Downloaded file 'OrderDetail.txt' from the server.
Downloaded file 'BillDetail.txt' from the server.
Disconnected from the ftp server.
> []
```

The user will have access to their order and bill details file.

```
serverSide.py  BillDetail.txt X
Users > warda1 > BillDetail.txt
1  ***
2
3  ['French Fries', 300]  Quantity= 3  Total= 900
4  ***
5
6  ['HP Laptop', 100000]  Quantity= 1  Total= 100000
7  ***
8
9  ['Computer Sheets', 150]  Quantity= 1  Total= 150
10 ***
11
12 ['Blue Jeans', 2000, 'M']  Quantity= 1  Total= 2000
13
```

```
serverSide.py  BillDetail.txt  OrderDetail.txt X
Users > warda1 > OrderDetail.txt
1  ***
2
3  ['French Fries', 300]
4  ***
5
6  ['HP Laptop', 100000]
7  ***
8
9  ['Computer Sheets', 150]
10 ***
11
12 ['Blue Jeans', 2000, 'M']
13
```



Then the user can go and select the payment option.

```
[SERVER] Welcome to KHAREEDLO. What would you like to do?
a.Shopping
b.Track Order
c.Feedback
d.Payment Option

To quit anytime, press q.
> d
[SERVER]
Select Payment method:
1.Cash-on-Delivery
2.Credit Card

> 2
[SERVER]
Payment done successfully.

> []
```

This will create a track file where the user can see when the products will be delivered to them.

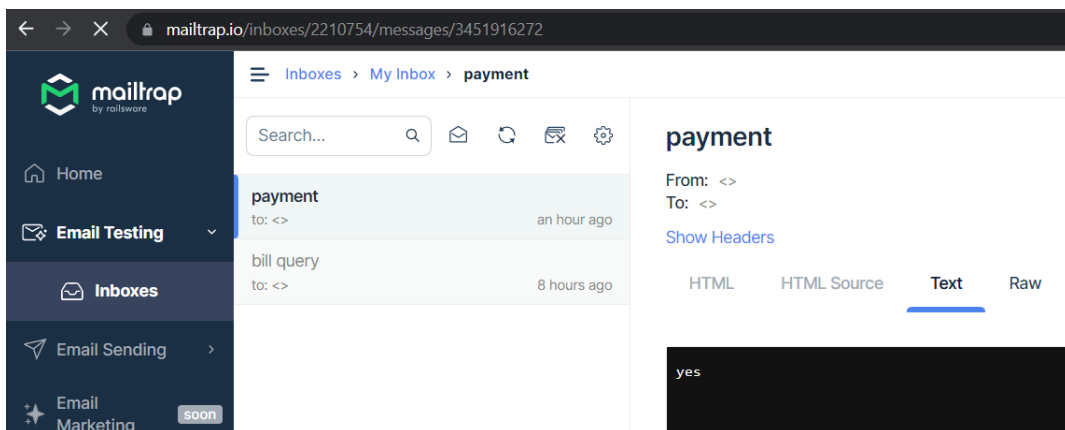
```
serverSide.py  TrackingDetails.txt X
Users > warda1 > TrackingDetails.txt
1  ***
2  Payment done through credit card.
3  Your order has been placed 2023-05-12 06:28:33.
4  It will be delivered in 3 working days.
```

We can directly access the track file using the main menu.

Other than this the user can also provide feedback to the admin using their main SMTP server.

```
PS C:\Users\warda\OneDrive\Desktop\CNTestProject> python clientSide.py
[CONNECTED] Client connected to server at 172.16.81.133:5566
[SERVER] Welcome to the server. Type 'signup' to create a new account or 'login' to log in
to an existing account.
> login
Type -feedback- if you are sure you want to send us a feedback.
> feedback
[SERVER] Do you want to do select again?
a.Shopping
b.Track Order
c.Feedback
d.Payment Option

>Enter your email: warda
>Enter your password: 1234
> Subject: payment
> Message: yes
Email sent successfully!
```



## **Conclusion**

The Chat-Enabled Shopping System provides a convenient and personalized shopping experience for customers by enabling real-time communication with sellers or customer service representatives. The system uses modern web technologies and protocols to ensure secure and efficient communication and file transfer. The system architecture and implementation details are discussed in this report, and we will conduct rigorous testing to ensure the system meets the requirements and provides a satisfactory user experience.

## References

1. R. Mishra and R. K. Singh, "A Chatbot-Based Shopping Assistance System Using IoT and Big Data Analytics," *IEEE Consumer Electronics Magazine*, vol. 8, no. 3, pp. 64-70, May 2019. doi: 10.1109/MCE.2019.2901928
2. S. Kumar, N. Sharma, and N. K. Singh, "Chatbot Based Online Shopping Assistant," 2018 5th International Conference on Signal Processing and Integrated Networks (SPIN), Noida, India, 2018, pp. 669-673. doi: 10.1109/SPIN.2018.8474091
3. A. Tiwari, "Chatbot Technology for E-Commerce Industry: A Review," 2019 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE), Srinagar, India, 2019, pp. 202-208. doi: 10.1109/ICCIKE47333.2019.9058607
4. J. Kim, J. Choi, and H. Kim, "Integrating Conversational Agents into E-commerce: Customer Perceptions and Satisfaction," *Journal of Retailing and Consumer Services*, vol. 49, pp. 58-69, Mar. 2019. doi: 10.1016/j.jretconser.2019.01.014
5. K. Abadie, "Online Shopping: Advantages, Disadvantages, and Tactics," *Journal of Business and Psychology*, vol. 17, no. 3, pp. 351-356, 2003. doi: 10.1023/A:1022883004350