Faculty of Graduate Studies

University of Kelaniya

Master of Information Technology

Python Based Newspaper Inventory Control System for EBC

PaperTrack

IMIT 53493

Object Oriented Programming

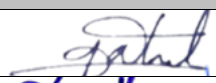
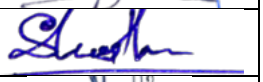
DECLARATION

We hereby declare that this report, titled " Python Based Newspaper Inventory Control System for EBC – PaperTrack", is the result of our group project work carried out for the modules IMIT 53493 – Object Oriented Programming under the Master of Information Technology degree program.

This report is based on our original work, except where references are clearly indicated. It has not been submitted previously, in whole or in part, for any other academic purpose or examination.

All members of the group have contributed to the completion of this project and the preparation of this report.

Names & Signatures of Group Members:

Member name	Index number	Signature
M.G.R.P. Yasasiri	FGS/MIT/2024/009	
G. Shasitha Madushanka Gamlath	FGS/MIT/2024/068	
S.A.I.A. Subhasinghe	FGS/MIT/2024/073	
E.M. Virani S. Ekanayake	FGS/MIT/2024/081	

Date: 27.09.2025

ABSTRACT

This project creates a Python based Stock Management system for Ekanayake Book City (EBC) in Polpithigama, Kuruneala who is handling newspapers, lotteries and stationaries. The existing manual process of order processing, advertisement handling and stock management is inefficient and time consuming which introduces human errors.

To address these inefficiencies the proposed system provides a Python based standalone system with Tkinter handling the user interfaces and MySQL to handle databases. This system automates the order processing, advertisement handling and stock management as well as reporting as the functional areas and addresses non functional areas like usability, security and reliability while establishing a foundation for future enhancements like online accessibility, mobile integration, automated notifications and payments gateway integration.

This project demonstrates the efficiency of using Object Oriented Programming concepts in streamlining a business operation while developing the team's technical proficiency.

ACKNOWLEDGMENT

We would take this as a great opportunity to express our sincere gratitude on every one who helped us in successfully completing the Python Based Newspaper Inventory Control System for EBC – PaperTrack which was carried out as a part of the Master of Information Technology degree program.

First and foremost, out heartfelt thank is extended to the lecture of the IMIT 53493 – Object Oriented Programming module, Senior Professor - Janaka Wijayanayake for giving us this opportunity to showcase what we have gained from the modules in the real-world situation.

We are also thankful to our course coordinator Dr. Dilani Wickramaarachchi and the admin staff of Faculty of Graduate studies, university of Kelaniya.

We also would like to thank the staff of Ekanayake Book City, Polpithigama for supporting us from the beginning of the process.

A special note of thank goes to all the groups members for the teamwork. Collaboration and dedication on completing the task.

Last but not least we would like to remember our families for their patient and encouragement during this time consuming and challenging phase of our academic journey.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

- **EBC** – Ekanayake Book City
- **OOP** – Object Oriented Programming
- **SDLC** – Software Development Life Cycle
- **GUI** – Graphical User Interface
- **OS** – Operating System
- **AI** – Artificial Intelligence
- **PC** – Personal Computer

CHAPTER ONE
INTRODUCTION

1.1 Introduction to the project

Ekanayake Book City (EBC) operates newspaper agencies and lotteries agencies while selling a wide variety of stationary Items including printed materials in Polpithigama, Kurunegala. EBC serves as the agent for multiple prominent newspaper companies by accepting newspaper and magazine orders and newspaper advertisements on behalf of them.

At present EBC handles the newspaper and magazine stock management part, order processing part and the newspaper advertisements handling part manually which has many drawbacks and consumes time as well as having a higher rate of human error.

In order to fulfill the requirement of the IMIT 53493 – Object Oriented Programming module, our team members propose a computer-based management system for EBC using Python to minimize the inconveniences caused by the manual system. In this system we mainly focus on the newspaper stock handling part and other aspects like lotteries, stationaries and other printed materials.

1.2 Objectives

The aim of the project is to develop a Python based standalone solution that will assist the EBC in handling of newspaper stocks, orders, customers and advertisements.

- Maintain the updated stocks of publications
- Accept and prepares the orders placed by the customers
- Generate the monthly newspaper bills (invoices) at the beginning/ end of the month
- Handle customers
- Handle newspaper advertisements
- Send email notifications accordingly
- Ensure the data security
- Ensure the privacy of data

1.3 Scope

- Stock handling system of publications (newspapers and magazines)
- Customer management system
- Order management system
- Invoice generation
- Notification sending via emails

CHAPTER TWO
BACKGROUND

Newspapers and magazines are supplied by different newspaper companies (Lakehouse, Wijaya, Sathara, Imashi etc..). publications are issued in different bases by the companies.

- Newspapers – issued daily or weekly
- Magazines – issued monthly

When the newspapers and magazines are received in the morning, the stock has to be updated immediately. After that the orders of the newspapers and magazines has to be processed.

Customers can place orders for the newspapers and magazines. Usually, personal orders are pre-paid, with the payment collected at the beginning of the month. Official orders (like government offices, schools and banks) are postpaid, with payment collected at the end of the month. When calculating the monthly amount for the official orders, it requires to concern on the weekends and public holidays which adds complexity to the calculation process. Sometimes, payment reminders are also need to be sent to them.

Some customers need to buy older newspapers as well. But EBC maintains only one month old stock with them.

At the end of the month the remaining stock of the newspapers and the magazines have to be returned with the records.

EBC accepts newspaper advertisements which are published in different newspapers and magazines. Some of the advertisement types available are like real estate, vehicle, marriage proposals and name changes etc.

When a customer wants to publish an advertisement, details like data of publication, target newspaper or magazine, content of the advertisement with some supporting documents and customer details are collected through printed forms. Then the total cost is calculated based in the number of words in the advertisement.

CHAPTER THREE
ANALYSIS AND DESIGN

3.1 Requirement Analysis

Requirement gathering and analyzing is one of the most important phases of the Software Development Life Cycle (SDLC).

A series of interviews was taken place with the management of the EBC and the team members. Further document sample analysis and observations of the working environment was also done in order to get a better understanding of the system background.

3.2 Basic Functional Requirements

Listed below are the core functions we have retracted from the gathered data.

- User registration and management of user profiles (Admin, Assistants)
- Stock handling of newspapers and magazines (Publications)
- Accepting and handling orders
- Handling advertisements
- Invoice generation (for orders and advertisements)
- Sending notifications for payments
- Report generation (stock details, payment details)

3.3 Non-Functional Requirements

Here are the identified nonfunctional requirements for the system.

- **Usability –** Interfaces of the system has to be user friendly with a minimal learning curve
- **Scalability –** Should have the ability to support the increasing number of customers and orders
- **Security –** The data collected from the customers and shared documents for advertisements should be properly secured with authentication and encryption methods

- **Reliability –** Downtime of the system should be minimal and the system must be available to the users in a stable way
- **Performance –** Response time should be fast
- **Accessibility –** System should be compatible across the different OS and hardware

3.4 Class Diagram

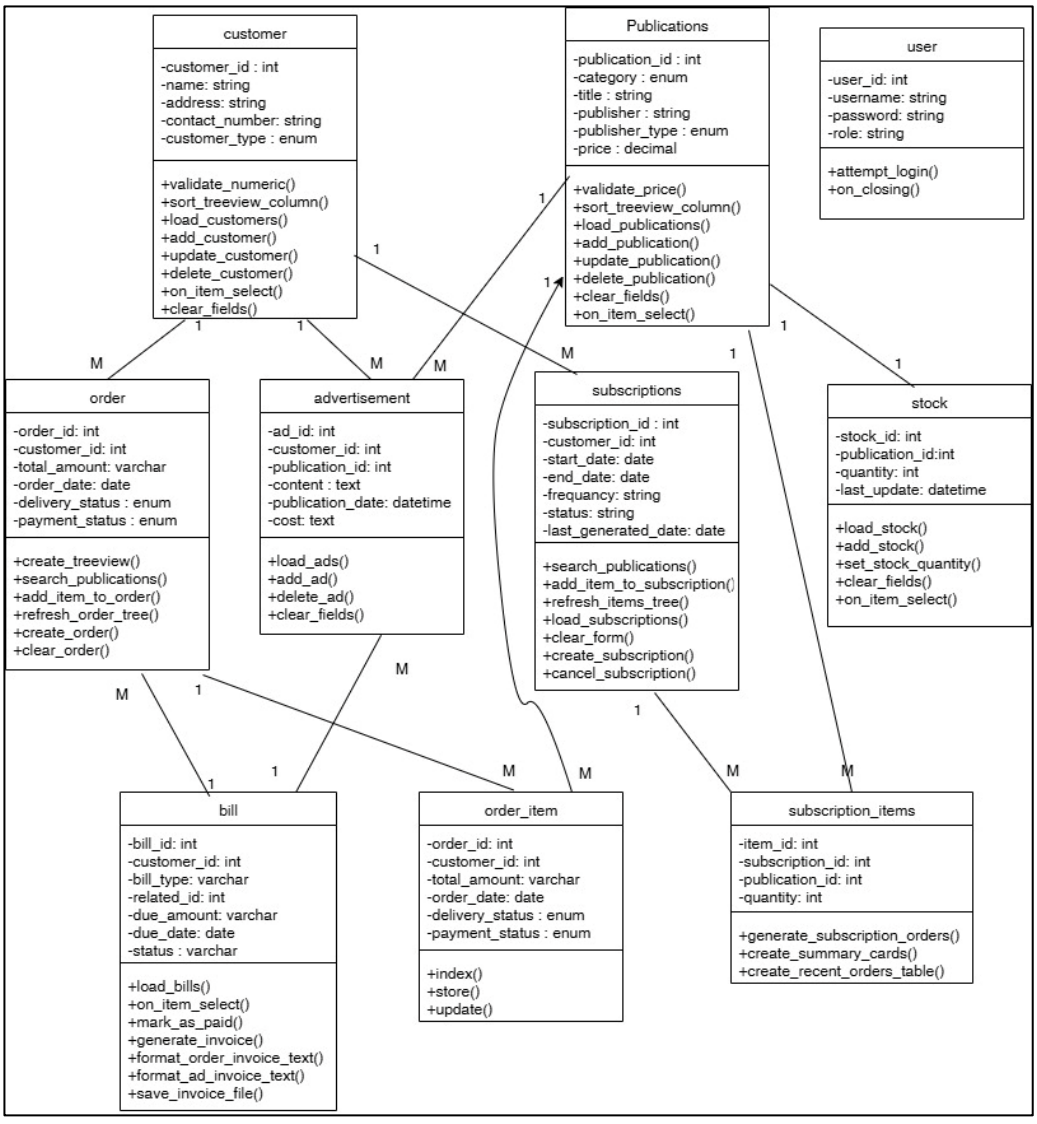


Figure 1 - Class Diagram

3.5 Sequence Diagrams

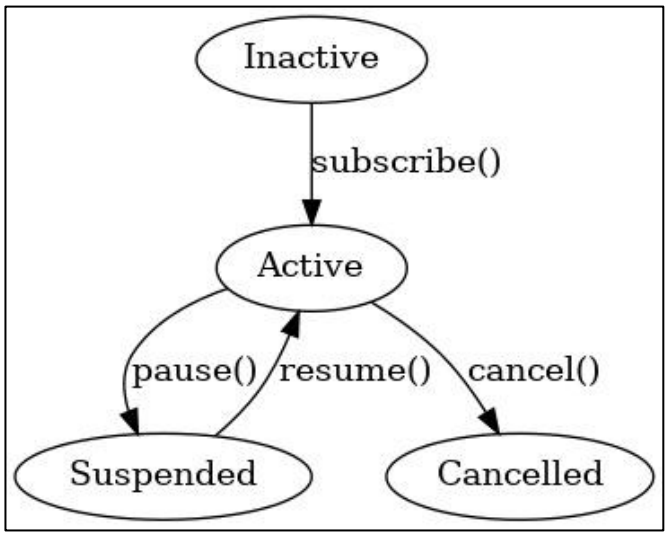


Figure 2 - Sequence Diagram for Order Processing

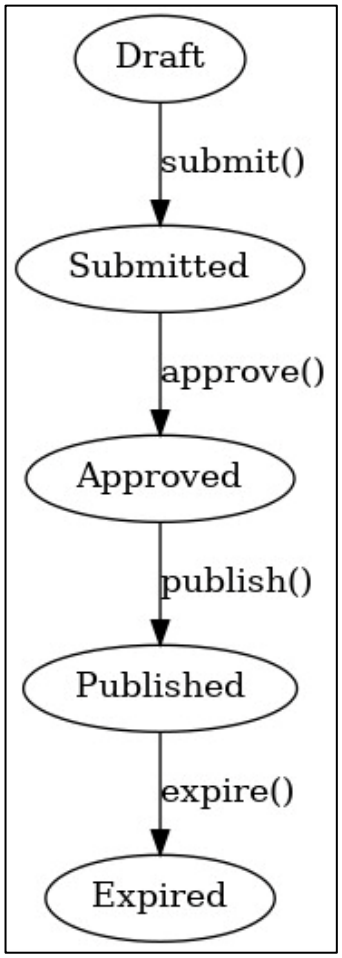


Figure 3 - Sequence Diagram for Subscription

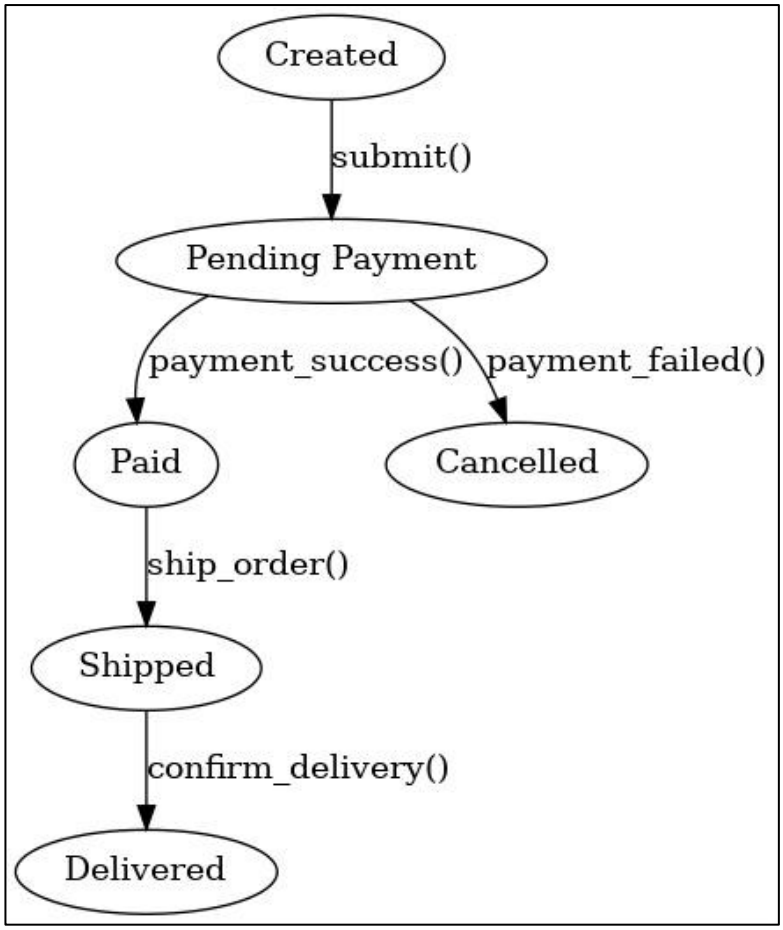


Figure 4 - Sequence Diagram for order Processing

3.6 State Diagram

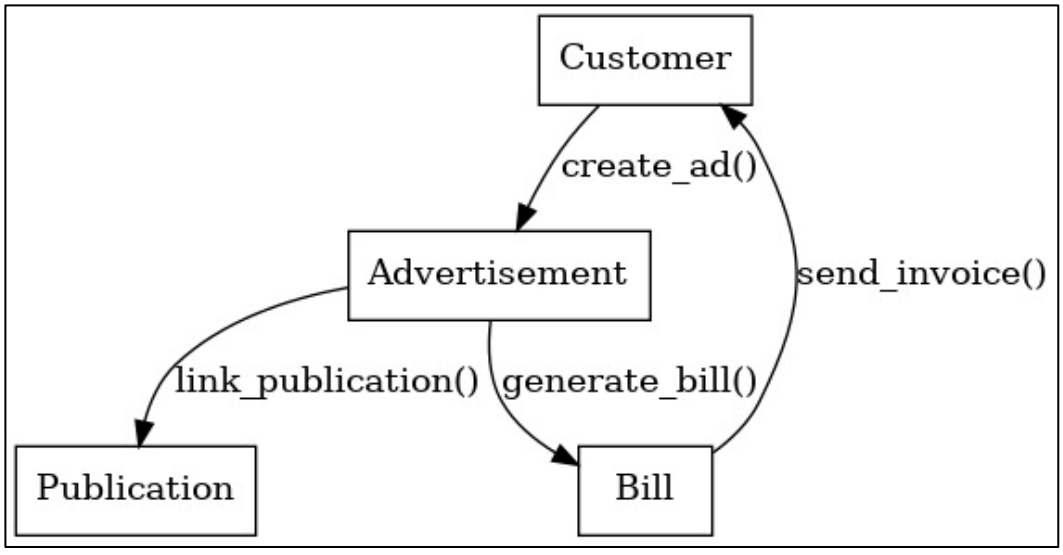


Figure 5 - State Diagram for Advertisement

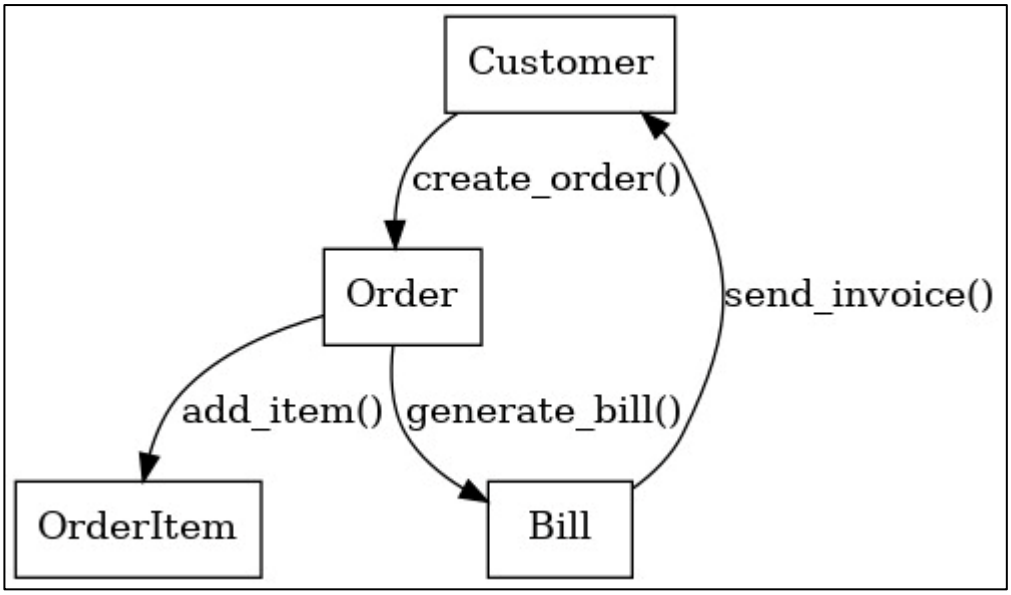


Figure 6 - State Diagram for Order

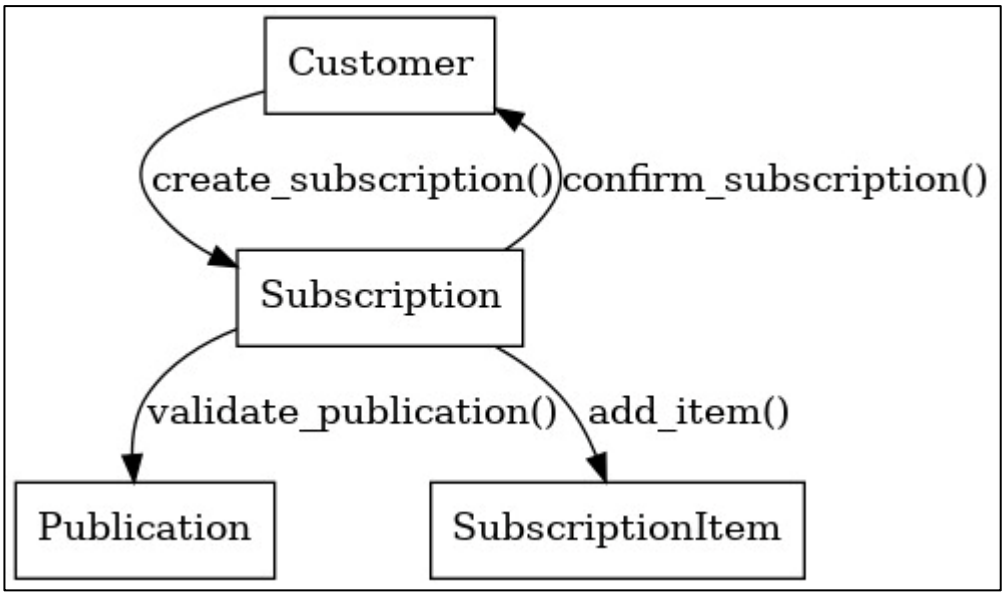


Figure 7 - State Diagram for Order

3.7 Use Case Diagram

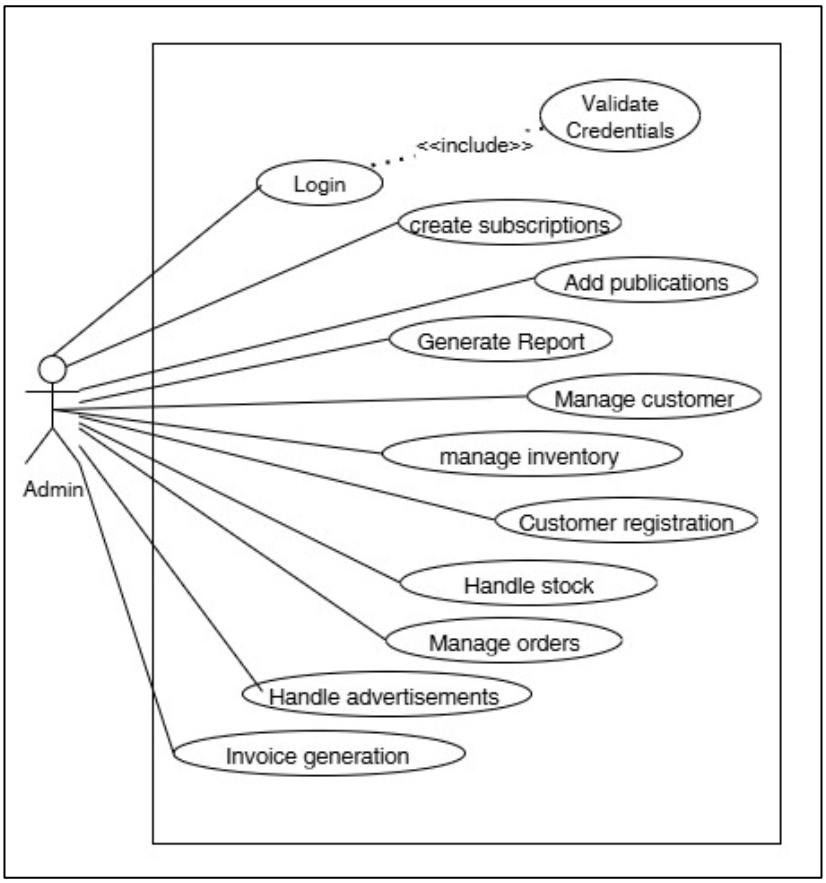


Figure 8 - Use Case Diagram

3.8 Design

3.8.1 Database Design

Database handling part of the developed systems was done using MySQL which is an open-source software.

Given below are the main tables used to store data with some dummy data.

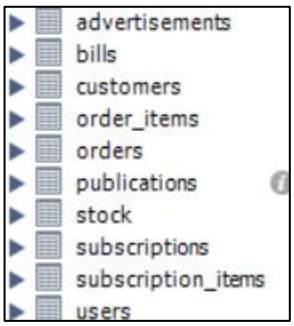


Figure 9 - Main Database Tables

ad_id	customer_id	publication_id	publication_date	content	cost
1	4	1	2025-09-21	Grand opening of Peter"s Cafe this Saturday! Vi...	260.00
2	5	6	2025-09-27	Looking for a skilled web developer. Contact us.	160.00
3	2	1	2025-09-23	This is my advertisement	80.00
NULL	NULL	NULL	NULL	NULL	NULL

Figure 10 - Advertisements

bill_id	customer_id	bill_type	related_id	due_amount	due_date	status
1	2	Order	1	1100.00	2025-09-21	Paid
2	3	Order	2	2000.00	2025-09-21	Unpaid
3	4	Advertisement	1	260.00	2025-09-21	Unpaid
4	5	Advertisement	2	160.00	2025-09-27	Paid
5	2	Order	3	1750.00	2025-09-23	Paid
6	4	Order	4	1750.00	2025-09-23	Unpaid
7	3	Order	5	225.00	2025-09-23	Unpaid
8	2	Advertisement	3	80.00	2025-09-23	Unpaid
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 11 - Bills

customer_id	name	address	contact_no	customer_type
2	John Doe	123, Main St, Colombo 07	0771234567	Postpaid
3	Jane Smith	456 Oak Ave, Kandy	0719876543	Prepaid
4	Peter Jones	789 Pine ln, Galle	0765551234	Postpaid
5	Samantha Lee	101 Maple Dr, Jaffna	0701122334	Prepaid
NULL	NULL	NULL	NULL	NULL

Figure 12 - Customers

order_item_id	order_id	publication_id	quantity	price_per_unit
1	1	1	10	80.00
2	1	2	2	150.00
3	2	5	5	400.00
4	3	4	5	350.00
5	4	4	5	350.00
6	5	2	1	150.00
7	5	6	1	75.00
NULL	NULL	NULL	NULL	NULL

Figure 13 - Order Items

order_id	customer_id	order_date	total_amount	delivery_status	payment_status
1	2	2025-09-21	1100.00	Delivered	Paid
2	3	2025-09-21	2000.00	Pending	Unpaid
3	2	2025-09-23	1750.00	Pending	Unpaid
4	4	2025-09-23	1750.00	Pending	Unpaid
5	3	2025-09-23	225.00	Pending	Unpaid
NULL	NULL	NULL	NULL	NULL	NULL

Figure 14 - Orders

publication_id	category	title	publisher	publish_type	price
1	Newspaper	Daily Mirror	Wijaya Newspapers	Daily	80.00
2	Newspaper	The Sunday Times	Wijaya Newspapers	Weekly	150.00
4	Newspaper	Reader"s Digest	RD Publications	Monthly	350.00
5	Magazine	Time Magazine	Time USA, LLC	Weekly	400.00
6	Newspaper	Daily News	Associated Newspapers of Ceylon	Daily	75.00
NULL	NULL	NULL	NULL	NULL	NULL

Figure 15 - Publications

stock_id	publication_id	quantity	last_updated
1	1	140	2025-09-21 20:47:16
2	2	197	2025-09-23 14:43:05
4	4	70	2025-09-23 10:37:32
5	5	45	2025-09-21 20:49:04
6	6	119	2025-09-23 14:43:05
NULL	NULL	NULL	NULL

Figure 16 - Stock

subscription_id	customer_id	start_date	end_date	frequency	status	last_generated_date
1	4	2025-09-23	2025-09-26	Daily	Cancelled	NULL
2	3	2025-09-23	2025-09-30	Daily	Active	2025-09-23
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 17 - Subscriptions

item_id	subscription_id	publication_id	quantity
1	1	6	5
2	2	2	1
3	2	6	1
NULL	NULL	NULL	NULL

Figure 18 - Subscription Items

user_id	username	password_hash	role
1	admin	admin	Admin
NULL	NULL	NULL	NULL

Figure 19 - Users

3.8.2 User Interfaces

Tkinter, which is a built-in Python module was used for creating the Graphical User Interfaces. It provides a simple way to create windows, buttons, text boxes and other GUI elements.

Here are the main user interfaces of the system.

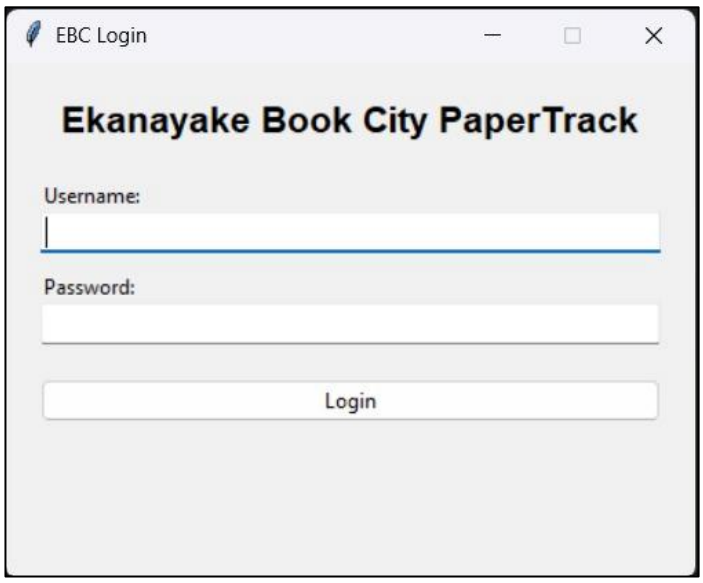


Figure 20 - Login Page

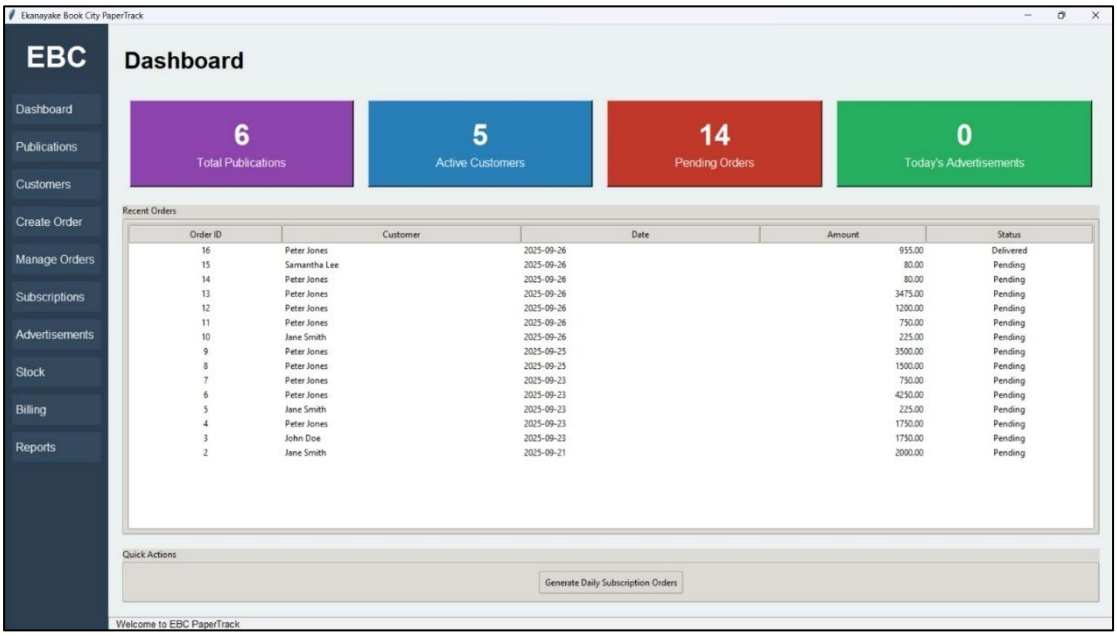


Figure 21 - Dashboard

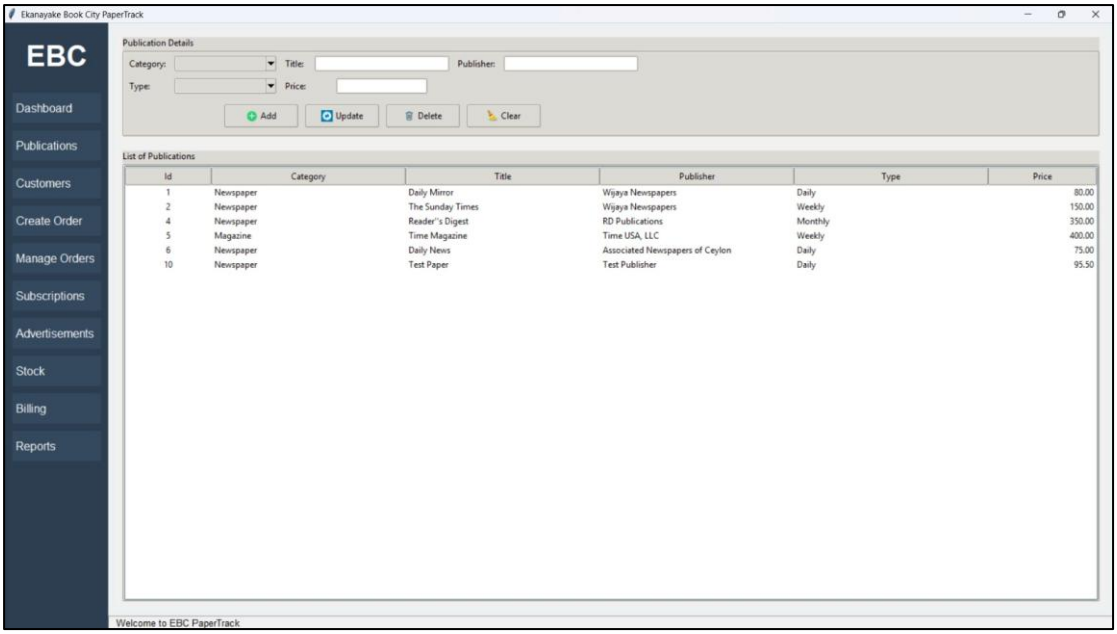


Figure 22 - Publications Page

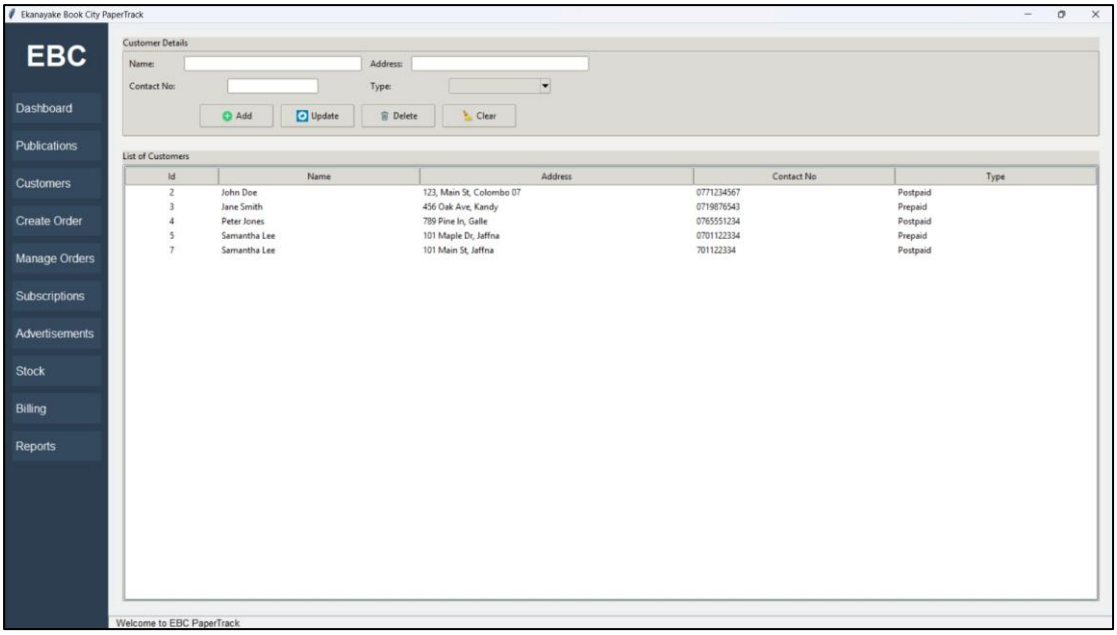


Figure 23 - Customers Page

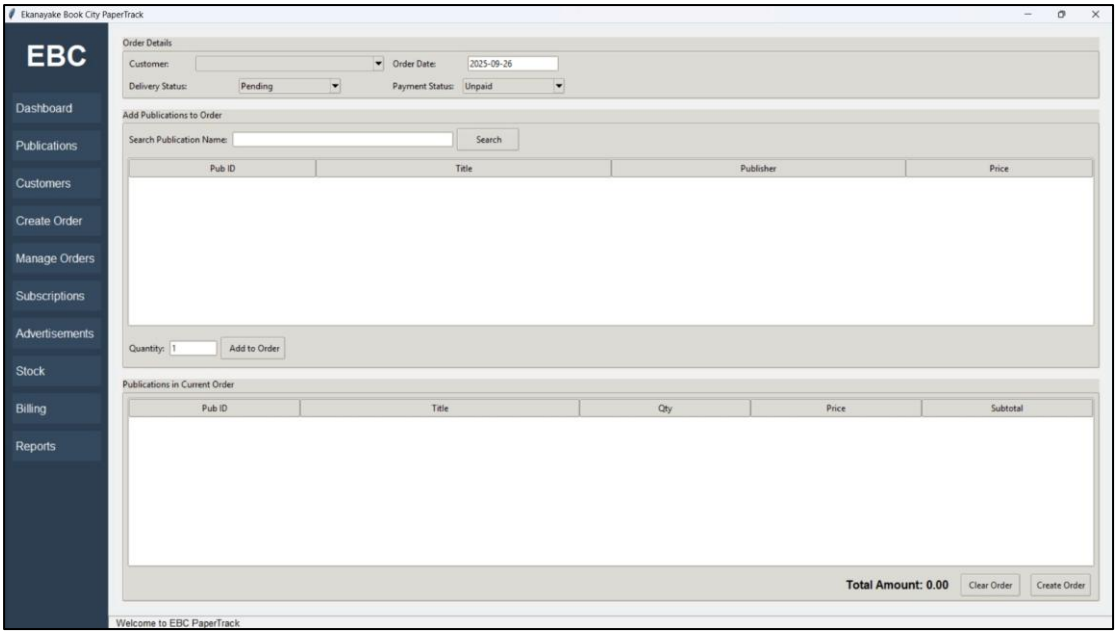


Figure 24 – Order Details Page

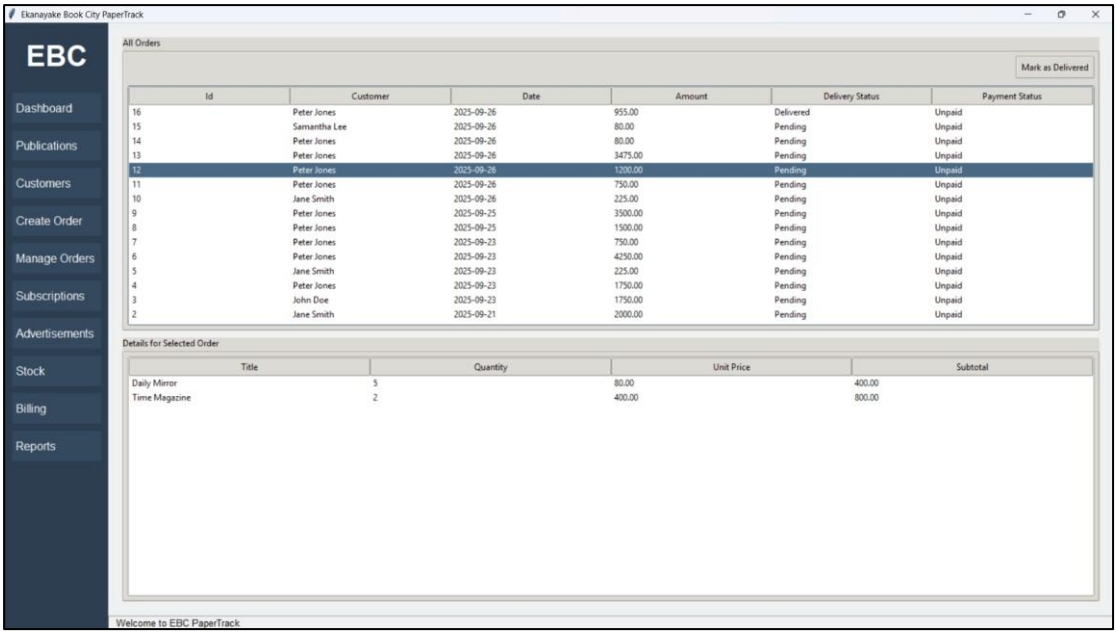


Figure 25 - All Orders Page

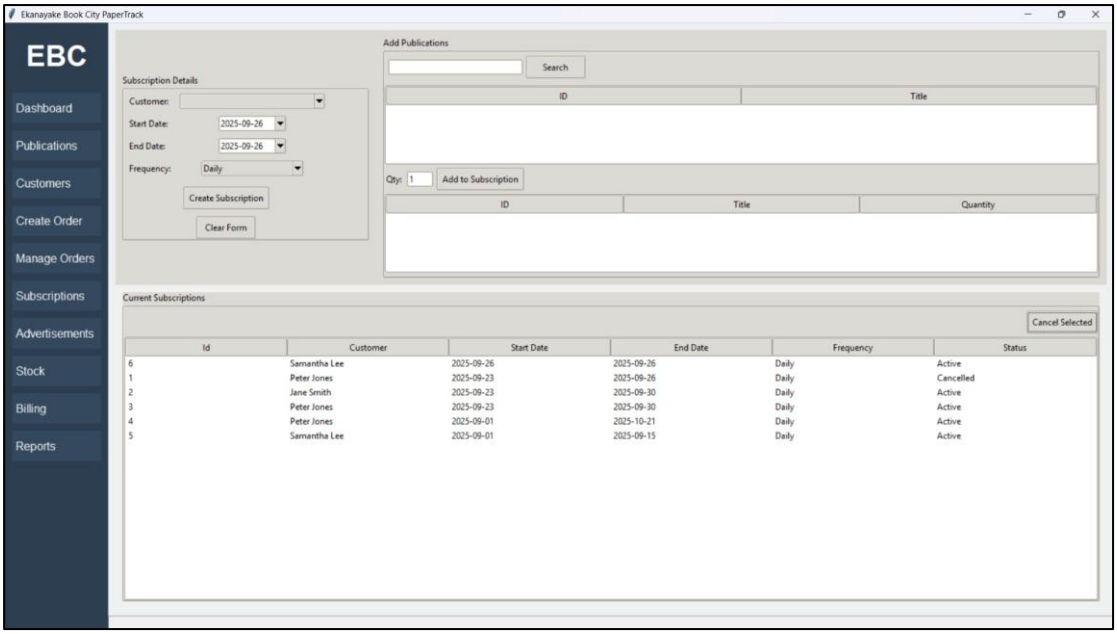


Figure 26 - Subscription Page

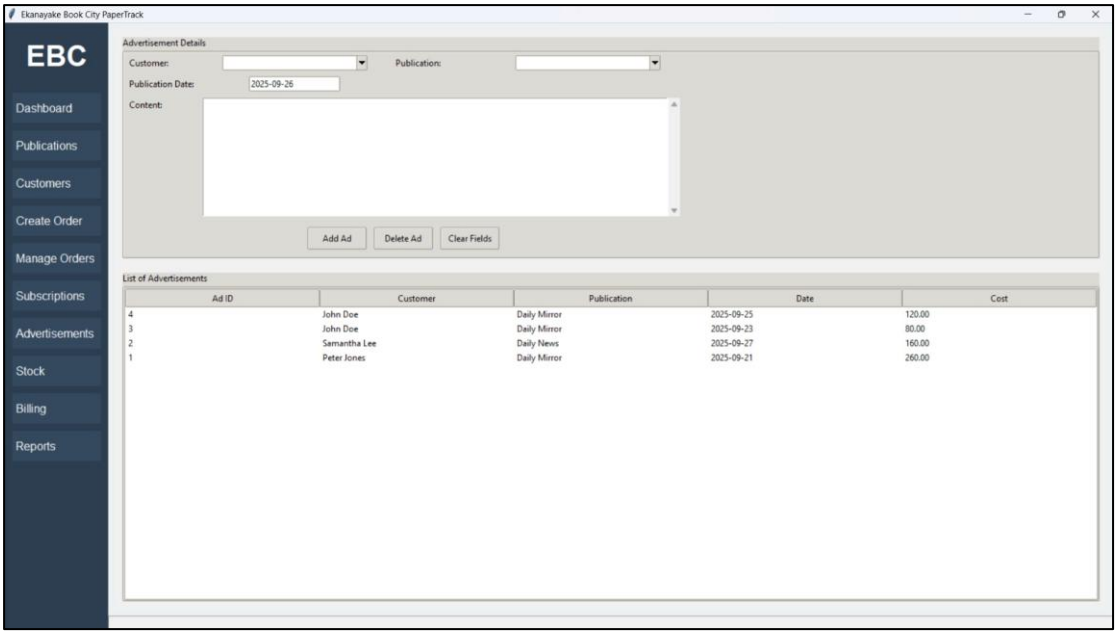


Figure 27 - Advertisement Page

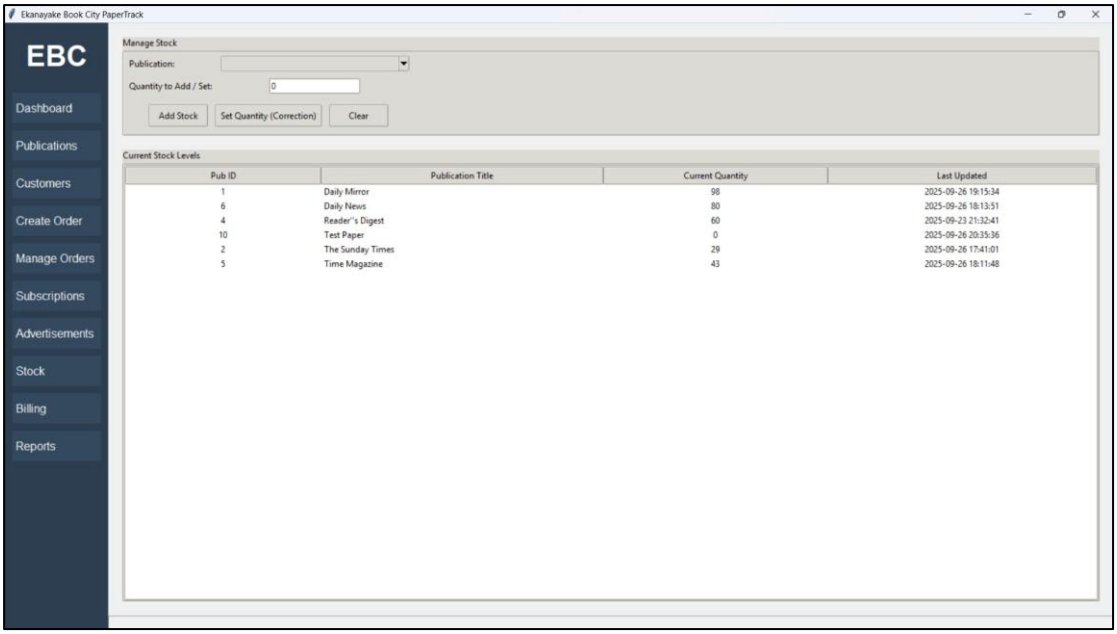


Figure 28 - Stock Page

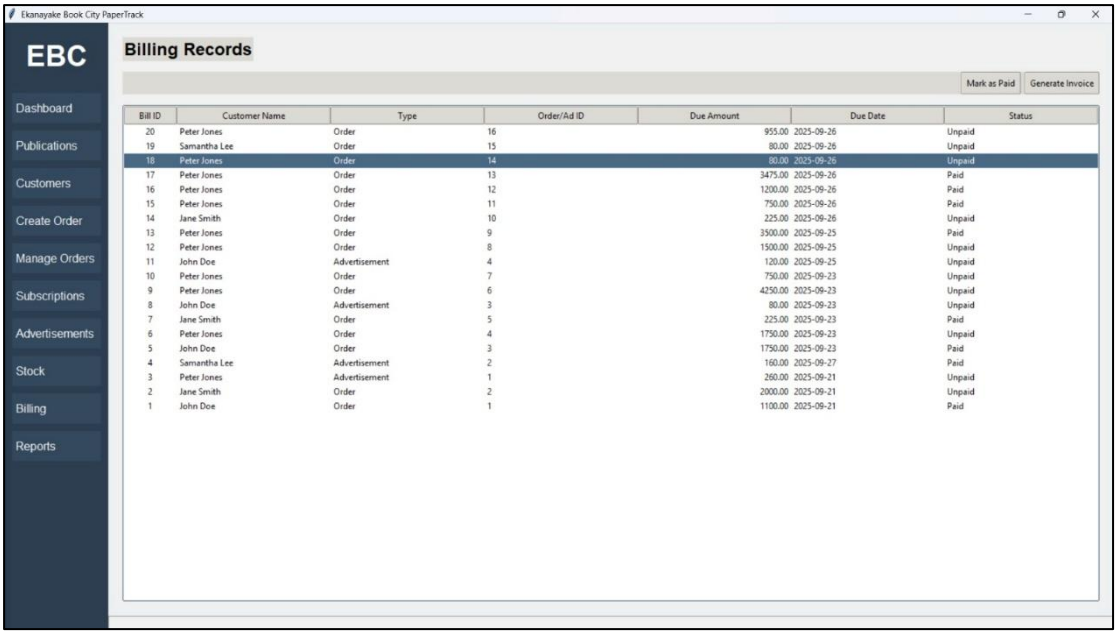


Figure 29 - Billing Page

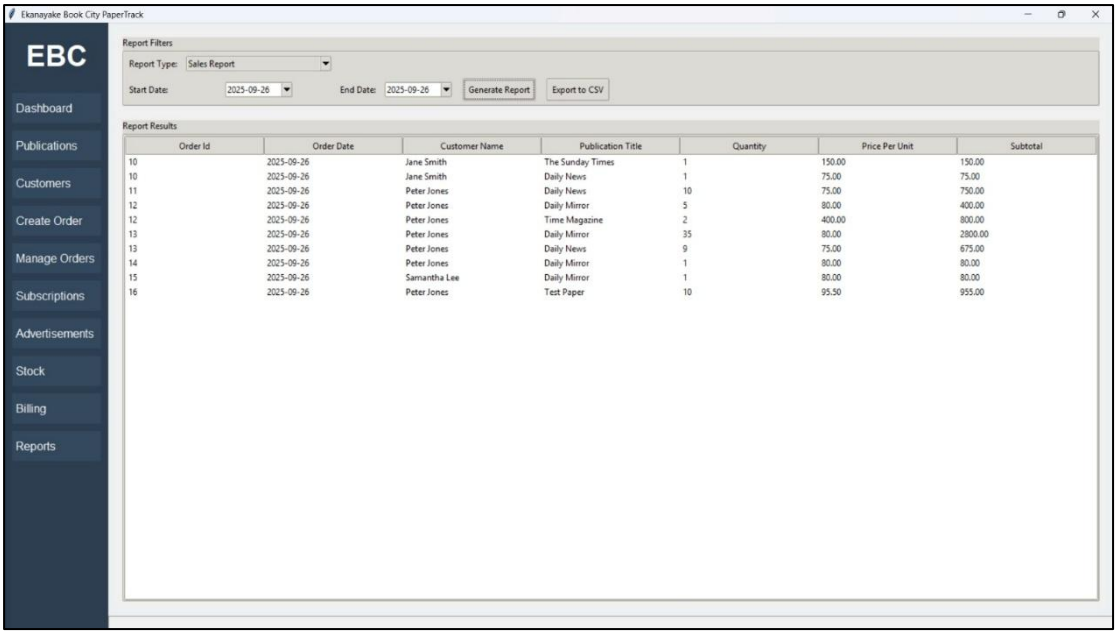


Figure 30 - Reports Page

```
======================================================================
                      Ekanayake Book City
                     ADVERTISEMENT INVOICE
======================================================================

                          *** PAID ***

Invoice #: 4
Ad ID:     2
Date:      2025-09-25 14:27:36

Bill To:
----------------------------------------------------------------------
Name:    Samantha Lee
Address: 101 Maple Dr, Jaffna
Contact: 0701122334

----------------------------------------------------------------------
Advertisement Details
----------------------------------------------------------------------
Publication:      Daily News
Publication Date: 2025-09-27

Content:
"Looking for a skilled web developer. Contact us."

----------------------------------------------------------------------
                                          COST:       160.00
======================================================================
                Thank you for your business!
======================================================================
```

Figure 31 - Invoice for Advertisements

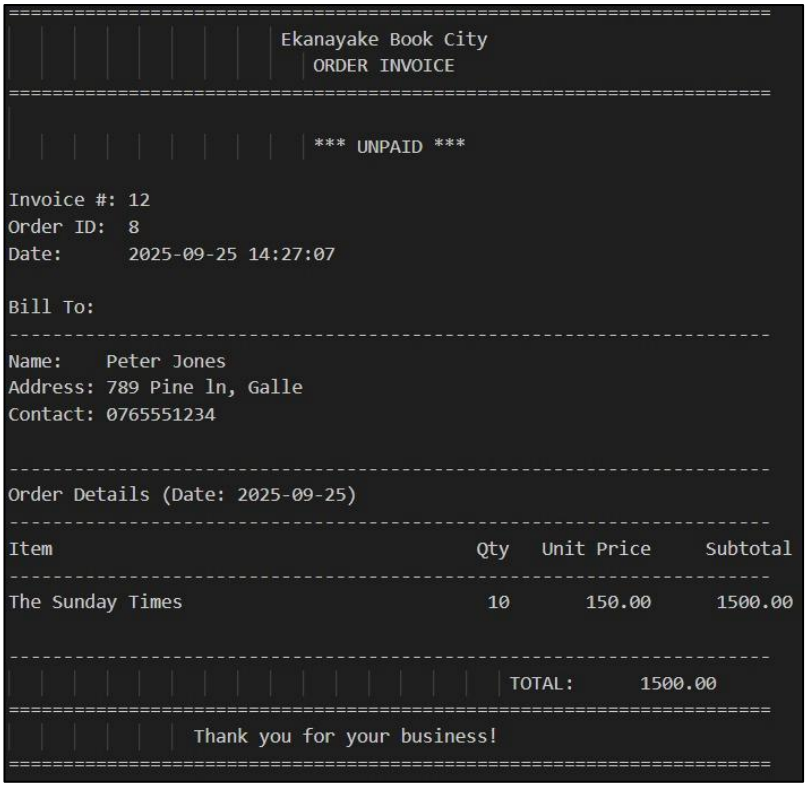


Figure 32 - Order Invoice

CHAPTER FOUR
IMPLEMENTATION

4.1 Limitations

PaperTrack offers an effective solution for the prevailing problems in the EBC environment. But we found there are some limitations as well in the system.

- Online inaccessibility

  We basically focused on fulfilling the requirements so the current version is not accessible through the internet or mobile devices.

- Handling only the newspaper and advertisement related tasks

  With the time concentration we had to narrow down the scope of the system only to newspapers and advertisements even though EBC deals with lotteries, stationaries and printed materials.

4.2 Technology Used

Component	Technology / Tool
Programming Language	Python
User Interface Design	Python (Tkinter)
Database	MySQL
Collaboration tools	Google Docs
	Zoom
Diagram design	Draw.io

4.3 Hardware Requirements

Following hardware requirements are used to implement this system.

- Desktop computer or laptop.
- Processor i3 or above and 2 GB RAM

CHAPTER FIVE
TESTING AND EVALUATION

5.1 Testing

The software system benefits from these testing techniques by producing high efficiency and efficiency and really accuracy.

Given below are the test cases we have used.

Module: Login & Authentication				
ID	**Test Objective**	**Test Steps**	**Expected Result**	**Type**
LOGIN-001	Verify successful login with correct credentials.	1. Launch main.py. 2. Enter username: admin. 3. Enter password: admin. 4. Click "Login".	The login window closes, and the main application dashboard opens.	Positive
LOGIN-002	Verify failed login with incorrect password.	1. Launch main.py. 2. Enter username: admin. 3. Enter password: wrongpassword. 4. Click "Login".	An error message "Invalid username or password" appears. The main application does not open. The password field is cleared.	Negative
LOGIN-003	Verify failed login with incorrect username.	1. Launch main.py. 2. Enter username: wrongadmin. 3. Enter password: admin. 4. Click "Login".	An error message "Invalid username or password" appears. The main application does not open.	Negative
LOGIN-004	Verify failed login with empty fields.	1. Launch main.py. 2. Leave username and password fields blank. 3. Click "Login".	An error message "Username and password are required" appears.	Negative
LOGIN-005	Verify that closing the login window exits the app.	1. Launch main.py. 2. Click the window's close (X) button.	The login window closes, and the application terminates without opening the main dashboard.	UI

Table 1 - Module: Login & Authentication

Module: Publications Management				
ID	**Test Objective**	**Test Steps**	**Expected Result**	**Type**
PUB-001	Verify adding a new publication successfully.	1. Navigate to the "Publications" page. 2. Fill in all fields with valid data (e.g., Category: Newspaper, Title: "Test Paper", Price: 95.50). 3. Click the "Add" button.	The status bar shows a success message. The new publication "Test Paper" appears at the top of the list. The entry fields are cleared.	Positive
PUB-002	Verify real-time validation for the Price field.	1. Navigate to the "Publications" page. 2. In the "Price" field, try to type letters (e.g., "abc").	The letters do not appear in the entry field. Only numbers and a single decimal point are allowed.	UI
PUB-003	Verify updating an existing publication.	1. Select "Test Paper" from the list. 2. Change its Price to 100.00. 3. Click the "Update" button.	The status bar shows an update success message. In the list, the price for "Test Paper" now shows 100.00.	Positive
PUB-004	Verify deleting a publication.	1. Select "Test Paper" from the list. 2. Click the "Delete" button. 3. Click "Yes" in the confirmation dialog.	The status bar shows a deletion success message. "Test Paper" is removed from the list.	Positive
PUB-005	Verify sorting by clicking a column header.	1. On the "Publications" page, click the "Title" header. 2. Click the "Title" header again.	The list sorts alphabetically by title (A-Z). The list sorts in reverse alphabetical order (Z-A).	UI
PUB-006	Verify adding a publication with a missing required field fails.	1. Fill in all fields except "Title". 2. Click the "Add" button.	An error pop-up message appears, stating that the title is required. The publication is not added.	Negative

Table 2 - Module: Publications Management

Module: Customer Management				
ID	**Test Objective**	**Test Steps**	**Expected Result**	**Type**
CUST-001	Verify adding a new postpaid customer.	1. Navigate to the "Customers" page. 2. Fill in all fields with valid data, selecting "Postpaid" as the type. 3. Click "Add".	The status bar shows a success message. The new customer appears in the list. Fields are cleared.	Positive
CUST-002	Verify real-time validation for the Contact No field.	1. On the "Customers" page, in the "Contact No" field, try to type letters.	The letters do not appear. Only numbers are allowed.	UI
CUST-003	Verify updating a customer's address.	1. Select an existing customer from the list. 2. Change their address. 3. Click "Update".	The status bar shows a success message. The customer's address is updated in the list.	Positive
CUST-004	Verify sorting by customer name.	1. Click the "Name" column header.	The list of customers is sorted alphabetically by name.	UI

Table 3 - Module: Customer Management

Module: Order Management & Stock				
ID	**Test Objective**	**Test Steps**	**Expected Result**	**Type**
ORD-001	Create a complete multi-item order.	1. Navigate to "Orders". 2. Select a customer. 3. Search for "Daily Mirror", add quantity 5. 4. Search for "Time Magazine", add quantity 2. 5. Verify both items appear in the "Current Order" list and the total is correct. 6. Click "Create Order".	The status bar shows an order creation success message. All fields on the order page are cleared.	Positive
ORD-002	Verify stock is updated after an order.	1. (Prerequisite: ORD-001) Navigate to the "Stock" page.	The quantity for "Daily Mirror" has decreased by 5. The quantity for "Time Magazine" has decreased by 2.	Positive
ORD-003	Verify a bill is created after an order.	1. (Prerequisite: ORD-001) Navigate to the "Billing" page.	A new bill for the correct customer and total amount from the order appears in the list with type "Order".	Positive
ORD-004	Verify creating an order with no customer fails.	1. Navigate to "Orders". 2. Add items to the order list. 3. Click "Create Order" without selecting a customer.	An error pop-up appears, stating a customer must be selected. The order is not created.	Negative

Table 4 - Module: Order Management & Stock

Module: Subscription Management				
ID	**Test Objective**	**Test Steps**	**Expected Result**	**Type**
SUB-001	Create a new daily subscription.	1. Navigate to "Subscriptions". 2. Select a customer, valid start/end dates, and "Daily" frequency. 3. Add one or more publications to the subscription. 4. Click "Create Subscription".	The status bar shows a success message. The new subscription appears in the list below.	Positive
SUB-002	Generate subscription orders when one is due.	1. (Prerequisite: SUB-001, ensure today's date is within the subscription range). 2. Navigate to "Dashboard". 3. Click "Generate Daily Subscription Orders".	A message box appears confirming 1 (or more) orders were generated. An order and a bill for the subscription appear on the "Orders" and "Billing" pages.	Positive
SUB-003	Verify no duplicate orders are generated.	1. (Prerequisite: SUB-002). 2. On the same day, click "Generate Daily Subscription Orders" again.	A message box appears stating "No subscription orders are due to be generated today." No new orders are created.	Positive

Table 5 - Module: Subscription Management

Module: Billing & Invoice				
ID	**Test Objective**	**Test Steps**	**Expected Result**	**Type**
BILL-001	Mark an unpaid bill as paid.	1. Navigate to "Billing". 2. Select a bill with "Unpaid" status. 3. Click "Mark as Paid" and confirm.	The status bar shows a success message. The bill's status in the list changes to "Paid".	Positive
BILL-002	Generate an invoice for an order.	1. Navigate to "Billing". 2. Select a bill of type "Order". 3. Click "Generate Invoice". 4. Save the file.	A save dialog appears. After saving, the status bar confirms it. The saved .txt file contains all customer and order item details, total amount, and payment status.	Positive
BILL-003	Generate an invoice for an advertisement.	1. Navigate to "Billing". 2. Select a bill of type "Advertisement". 3. Click "Generate Invoice" and save the file.	The saved .txt file contains all customer and advertisement details, cost, and payment status.	Positive

Table 6 - Module: Billing & Invoice

Module: Reporting				
ID	**Test Objective**	**Test Steps**	**Expected Result**	**Type**
RPT-001	Generate and export a Stock Level Report.	1. Navigate to "Reports". 2. Select "Stock Level Report" from the dropdown. 3. Click "Generate Report". 4. Click "Export to CSV" and save the file.	The stock levels are displayed in the on-screen table. The exported CSV file contains the correct headers and data matching the on-screen report.	Positive
RPT-002	Generate a Sales Report with a date filter.	1. Navigate to "Reports". 2. Select "Sales Report". 3. Choose a date range that includes known orders. 4. Click "Generate Report".	The on-screen table populates with sales data only from within the selected date range.	Positive

Table 7 - Module: Reporting

CHAPTER SIX
CONCLUSION

6.1 Lessons Learnt

During the experiment, a number of significant lessons were discovered:

- **Teamwork**
  Productivity was greatly increased by regular communication and task distribution.
- **User Centric Design**

  Developing significant and pertinent features required continuous input from user of the system

- **Technical Proficiency**
  Members of the team improved their programming abilities

6.2 Future Improvements

Currently, the system only handles the stock management part of the newspapers and magazines (publications) along with the advertisements handling part and the system is an offline standalone system.

Other aspects like handling of the lotteries, stationaries and other printed materials should be included as future improvements to the systems.

The system should be accessible via internet and should be able to use over mobile devices as well.

With the time limitation some of the features could not be implemented as per our plan, email notification is one of them. We wish to implement it in the next version.

Future versions will include payments through credit and debit cards as well as payment subscriptions, AI chatbot and a mobile application.

Currently we install the system to only one PC and with the growing number of customers, it needs to be installed into multiple PCs as well.

6.3 Difficulties Encountered and how they were solved

- **Clarity of Requirements:**
  At first, it was difficult to collect user expectations. Interviews and document inspections along with observations of the process could address this issue.

- **Time Restrictions:**
  Personal and academic schedules occasionally clashed. The team was able to maintain focus through efficient task distribution and frequent meetings.