K.RAMAKRISHNAN
COLLEGE OF ENGINEERING
An Autonomous Institution
Permanently Affiliated to Anna University Chennai, Approved by AICTE New Delhi,
ISO 9001:2015, 14001:2015 certified institution, Accredited by NBA and with A grade by NAAC
Samayapuram, Tiruchirappalli – 621 112, Tamilnadu, India.

Revenue Collection And Tracking System

A PROJECT REPORT

Submitted by

SILAMBU K(8115U23ME044)

in partial fulfillment of requirements for the award of the course

MGB1201 – PYTHON PROGRAMMING

In

DEPARTMENT OF MECHANICAL ENGINEERING

K.RAMAKRISHNAN COLLEGE OF ENGINEERING

(An Autonomous Institution affiliated to Anna University, Chennai Approved by AICTE,New Delhi)

SAMAYAPURAM-621 112

DECEMBER 2024

K. RAMAKRISHNAN COLLEGE OF ENGINEERING

(Autonomous Institution affiliated to Anna University, Chennai)

TRICHY-621 112

BONAFIDE CERTIFICATE

Certified that this project report on **"REVENUE COLLECTION AND TRACKING SYSTEM"**is the bonafide work of **SILAMBU K (8115U23ME044)** who carried out the project work during the academic year 2024 - 2025 under my supervision.

SIGNATURE
Dr. T. M. NITHYA, M.E.,Ph.D.,
HEAD OF THE DEPARTMENT
ASSOCIATE PROFESSOR
Department of CSE
K.Ramakrishnan College of Engineering (Autonomous)
Samayapuram–621112.

SIGNATURE
Mrs.S.RAJESWARI M.E.
SUPERVISOR
ASSISTANT PROFESSOR
Department of CSE
K.Ramakrishnan College of Engineering (Autonomous)
Samayapuram–621112.

Submitted for the End Semester Examination held on…….

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION

I declare that the project report on **"Revenue Collection And Tracking System"** is the result of original work done by us and best of our knowledge, similar workhas not been submitted to **"ANNA UNIVERSITY CHENNAI"** for the requirement of Degree of **BACHELOR OF ENGINEERING**. This project report is submitted on the partial fulfilment of the requirement of the completion of the course **MGB1201 –**

PYTHON PROGRAMMING

.

Signature

_____

SILAMBU K

Place: Samayapuram

Date:

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution "**K.Ramakrishnan College of Engineering (Autonomous)**", for providing us with the opportunity to do this project.

I glad to credit honourable chairman **Dr. K. RAMAKRISHNAN**, **B.E.,** for having provided for the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director **Dr. S. KUPPUSAMY, MBA, Ph.D.,** for forwarding to our project and offering adequateduration in completing our project.

I would like to thank **Dr. D. SRINIVASAN, B.E, M.E., Ph.D.,**Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Dr. T. M. NITHYA, M.E.,Ph.D.,** Head of the department, **COMPUTER SCIENCE AND ENGINEERING** for providing her encourage pursuing this project.

I express our deep expression and sincere gratitude to our project supervisor **Mrs.S.RAJESWARI M.E.,** Department of **COMPUTER SCIENCE AND ENGINEERING,** for his incalculable suggestions, creativity, assistance and patiencewhich motivated us to carry out this project.

I render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VISION OF THE INSTITUTION

To achieve a prominent position among the top technical institutions

MISSION OF THE INSTITUTION

M1: To bestow standard technical education par excellence through state of the art infrastructure, competent faculty and high ethical standards.

M2: To nurture research and entrepreneurial skills among students in cutting edge technologies.

M3: To provide education for developing high-quality professionals to transform the society.

VISION OF THE DEPARTMENT

To create eminent professionals of Computer Science and Engineering by imparting quality education.

MISSION OF THE DEPARTMENT

M1: To provide technical exposure in the field of Computer Science and Engineering through state of the art infrastructure and ethical standards.

M2: To engage the students in research and development activities in the field of Computer Science and Engineering.

M3: To empower the learners to involve in industrial and multi-disciplinary projects for addressing the societal needs.

PROGRAM EDUCATIONAL OBJECTIVES (PEOS)

Our graduates shall

PEO1: Analyse, design and create innovative products for addressing social needs.

PEO2: Equip themselves for employability, higher studies and research.

PEO3: Nurture the leadership qualities and entrepreneurial skills for their successful career.

PROGRAM OUTCOMES

Engineering students will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write

11. effective reports and design documentation, make effective presentations, and give and receive clear instructions.
12. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
13. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSOs)

- **PSO1:** Apply the basic and advanced knowledge in developing software, hardware and firmware solutions addressing real life problems.

- **PSO2:** Design, develop, test and implement product-based solutions for their career enhancement.

ABSTRACT

The Revenue Management System is an efficient Python-based application designed to manage and track financial revenues by leveraging user input and data processing techniques. Featuring multiple functionalities such as adding new revenue entries, displaying all revenue entries, calculating total revenue, finding revenue for a specific date, exporting revenues to a file, and importing revenues from a file, the system offers users a comprehensive and interactive experience through a simple selection process. Upon choosing an operation, users can either input new revenue data, view detailed revenue information, or perform specific calculations, ensuring effective financial tracking and management. By incorporating robust error handling and userfriendly input mechanisms, the application guarantees accurate and reliable revenue tracking, providing a seamless experience with each use**.** The intuitive user interface prompts users to select their desired operation, resulting in an immediate and efficient revenue management process. This revision aligns the description with the functionalities provided by the specific functions in your revenue management application

ABSTRACT WITH POs AND PSOs MAPPING

ABSTRACT	POs MAPPED	PSOs MAPPED
The Revenue Management System is an efficient Python-based application designed to manage and track financial revenues by leveraging user input and data processing techniques. Featuring multiple functionalities such as adding new revenue entries, displaying all revenue entries, calculating total revenue, finding revenue for a specific date, exporting revenues to a file, and importing revenues from a file, the system offers users a comprehensive and interactive experience through a simple selection process. Upon choosing an operation, users can either input new revenue data, view detailed revenue information, or perform specific calculations, ensuring effective financial tracking and management.	**PO1 - 1** **PO2 -1** **PO3 -1** **PO12-2**	**PSO1 - 3**

SUPERVISOR **HEAD OF THE DEPARTMENT**

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Objective

In Creating a Revenue Management System in Python involves designing a simple yet effective tool to manage revenue records. This system allows users tperform operations such as adding new revenue entries, viewing all entries, calculating total revenue, searching for specific revenue entries by date, exporting revenue data to a file, and importing revenue data from a file. Byleveraging Python's file handling and basic data manipulation capabilities, wecan create a system that ensures data persistence even after the program is closed**.**

1.2 Overview

The Revenue Collection and Tracking System (RCTS) is an integrated platform designed to manage the entire process of revenue generation, collection, and tracking for businesses, government agencies, and organizations. It simplifies the complex task of handling payments, invoices, taxes, and financial transactions. The system automates key processes, ensuring accurate tracking of all revenue-related activities, which reduces human errors and enhances operational efficiency. With RCTS, users can easily issue invoices, collect payments, and monitor the status of outstanding balances.

The system offers real-time data access, enabling businesses and organizations to track revenues as they are collected, providing up-to-date financial reporting and insights flow management.

1.3 Python Programming Concepts

Database Management

1. Relational Databases: SQL-based databases (like MySQL, PostgreSQL) are commonly used to store revenue data, including transactions, customers, invoices, payments, etc. Proper database schema design is important for organizing tables and relationships (e.g., between customers, invoices, and payments).
2. Normalization: Ensuring that the database tables are normalized helps in avoiding redundancy and improving data integrity.

2.CRUD

Operations (Create, Read, Update, Delete)

1.Create: Adding new records, such as new transactions, invoices, or customers.

2.Read: Retrieving records for display or processing, such as viewing total revenue or filtering transactions by date.

3.Update: Modifying records, such as updating a payment status or correcting an invoice.

4.Delete: Removing data when necessary, like deleting erroneous or outdated transactions.

Authentication and Authorization

1.User Authentication: Ensuring that only authorized users (e.g., accountants, managers) can access sensitive revenue data.

2.Role-Based Access Control (RBAC): Implementing different levels of access control, where users may have different permissions based on their roles .

CHAPTER 2

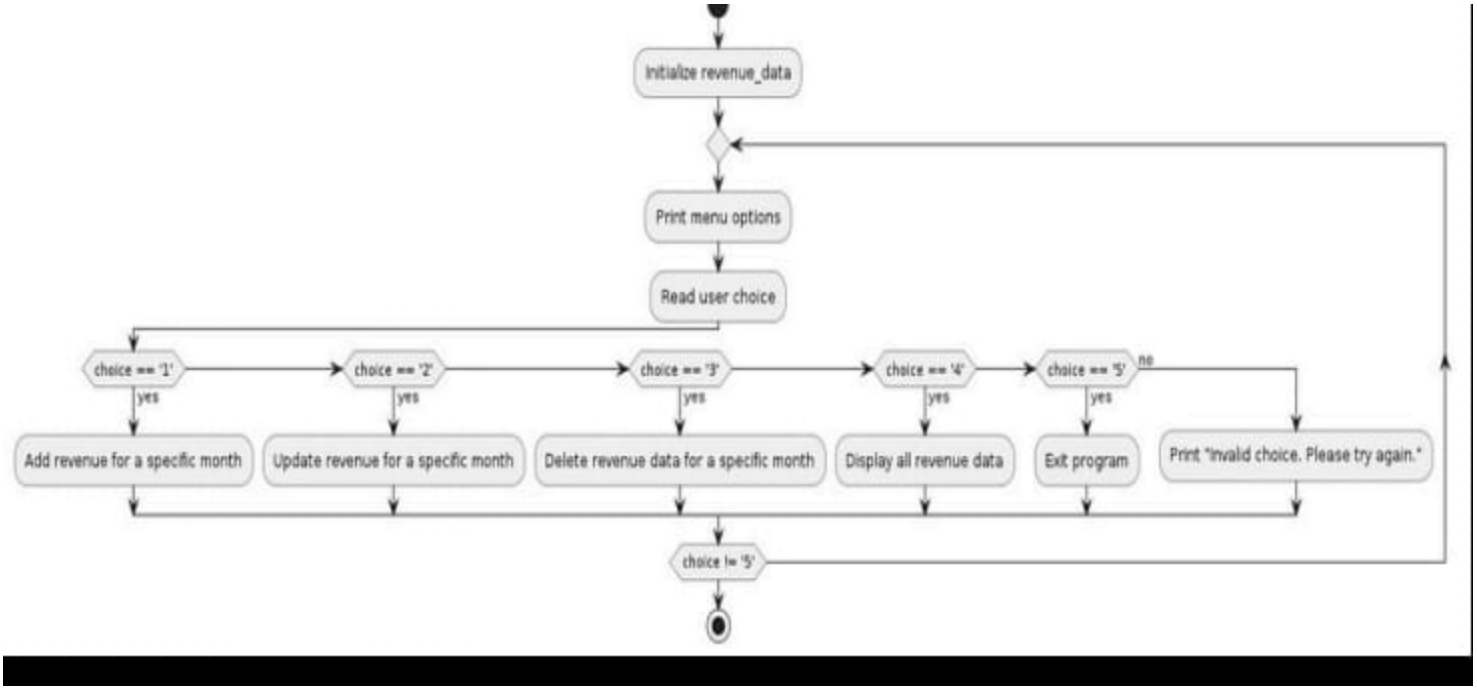
PROJECT METHODOLOGY

2.1Proposed Work

System architecture is the conceptual framework that defines the structureand behavior of a software system. It outlines the interactions between different components and modules within the system. In the context of therevenue management system, the architecture delineates how functionssuch as adding revenue entries, displaying revenues, and calculating totalrevenue are organized and interconnected.

It provides a blueprint for understanding how the system operates and evolves.
By defining the relationships and dependencies between various functions, system architecture facilitates efficient development, maintenance, and scalabilityof the software. Ultimately, it ensures that the system meets its functional requirements while remaining robust and adaptable.

2.2 Block Diagram



CHAPTER 3

MODULE DESCRIPTION

3.1 Module 1: Revenue Collection AndTracking System

1. Payment Collection Module

Purpose: This module is responsible for managing the collection of payments from customers or clients.

Key Features: Payment Gateway Integration: Supports various payment methods (credit cards, bank transfers, digital wallets, etc.).

Multiple Payment Channels: Supports both online (web/mobile) and offline (in-person) payment methods.

2.Revenue Tracking and Monitoring Module

Purpose: To track and monitor all payments and revenue generated in real-time.

Key Features:

Transaction Monitoring: Tracks the status of each payment (pending, completed, failed, etc.).

Revenue Reporting: Provides real-time reports on total revenue, outstanding payments, and payment trends.

Audit Trails: Logs all transactions for future auditing, ensuring transparency and security.

3.3Module :Features of Python

Payment Collection & Integration

3.Payment Gateway Integration: Python can integrate with various payment gateways such as Stripe, PayPal, or Razorpay using their SDKs and APIs to process payments. Libraries like requests and Stripe can be used to securely handle online transactions.

Multi-Channel Payment Support: Python can handle both online (web or mobile) and offline payment methods. For example, integrating with systems like `QR Code` generators (usingqrcode library) to facilitate offline payments.

3.4 Payment Reconciliation

- **Automated Reconciliation**: Python can match received payments against invoices by querying databases or using the `pandas` library to perform data operations, comparing payment records with pending invoices.
- **Bank Integration**: Python can connect to banking APIs or use scraping techniques (e.g., `BeautifulSoup` or `Selenium`) to reconcile payments with bank statements for a seamless financial workflow.
- **Exception Handling**: If discrepancies arise (e.g., mismatched payments), Python's error handling and exception mechanisms can identify and flag errors for further investigation.

CHAPTER 4

RESULTS AND DISCUSSION

PROGRAM

```
def add_revenue(revenues):
    while True:
        try:
            date = input("Enter date (YYYY-MM-DD): ")
            amount = float(input("Enter amount: "))
            if date in revenues:
                revenues[date] += amount
            else:
                revenues[date] = amount
            print("Revenue entry added successfully.")
            break
        except ValueError:
            print("Invalid input. Please enter a valid amount.")

# Function to display all revenue entries
def display_revenues(revenues):
    if not revenues:
        print("No revenue entries.")
    else:
        print("Date\t\tAmount")
```

```
        print("Date\t\tAmount")
        for date, amount in revenues.items():
            print(f"{date}\t{amount:.2f}")

# Function to calculate total revenue
def calculate_total_revenue(revenues):
    return sum(revenues.values())

# Function to find revenue for a specific date
def find_revenue_by_date(revenues):
    date = input("Enter date (YYYY-MM-DD): ")
    if date in revenues:
        print(f"Revenue on {date}: {revenues[date]:.2f}")
    else:
        print(f"No revenue found for {date}")

# Function to export revenues to a file
def export_revenues(revenues):
    filename = input("Enter filename to export revenues: ")
    try:
        with open(filename, "w") as file:
```

```
            with open(filename, "w") as file:
                for date, amount in revenues.items():
                    file.write(f"{date},{amount:.2f}\n")
            print("Revenues exported successfully.")
        except IOError:
            print("Error occurred while exporting revenues.")

    # Function to import revenues from a file
    def import_revenues(revenues):
        filename = input("Enter filename to import revenues: ")
        try:
            with open(filename, "r") as file:
                for line in file:
                    date, amount = line.strip().split(",")
                    revenues[date] = float(amount)
            print("Revenues imported successfully.")
        except (IOError, ValueError):
            print("Error occurred while importing revenues.")

    # Main function
    def main():
```

```
        revenues = {}  # Dictionary to store revenues
        while True:
            print("\nOptions:")
            print("1. Add Revenue")
            print("2. Display Revenues")
            print("3. Calculate Total Revenue")
            print("4. Find Revenue by Date")
            print("5. Export Revenues to File")
            print("6. Import Revenues from File")
            print("7. Exit")

            choice = input("Enter your choice: ")
            if choice == "1":
                add_revenue(revenues)
            elif choice == "2":
                display_revenues(revenues)
            elif choice == "3":
                print("Total Revenue:", calculate_total_revenue(revenues))
            elif choice == "4":
                find_revenue_by_date(revenues)
```

```
81          elif choice == "5":
82              export_revenues(revenues)
83          elif choice == "6":
84              import_revenues(revenues)
85          elif choice == "7":
86              print("Exiting program.")
87              break
88          else:
89              print("Invalid choice. Please try again.")
90
91  # Run the program
92  main()
```

OUTPUT

```
$ python CTP28132.py ID

Options:
1. Add Revenue
2. Display Revenues
3. Calculate Total Revenue
4. Find Revenue by Date
5. Export Revenues to File
6. Import Revenues from File
7. Exit
```

CHAPTER 5

CONCLUSION

The development and implementation of a **Revenue Collection and Tracking System** (RCTS) is essential for any organization, business, or government body looking to streamline its financial processes, improve transparency, and enhance accountability. The system allows for efficient and accurate tracking of revenues, whether they stem from taxes, sales, services, or other income streams. Through automation and centralization, an RCTS minimizes human error, reduces fraud, and ensures timely reporting and compliance with financial regulations.

Key takeaways from the implementation of an RCTS include:

1. **Improved Efficiency**: By automating revenue collection processes and integrating data sources, the system ensures quicker processing times and reduces administrative burdens.
2. **Enhanced Transparency**: Real-time tracking allows stakeholders to access up-to-date information on revenue, fostering a culture of openness and trust.
3. **Better Decision-Making**: With data analytics and reporting features, decision-makers are equipped with actionable insights that help them make informed financial decisions and forecast future revenues more accurately

REFERENCES:

1.**Flask Web Development:** Developing Web Applications

- This book focuses on using Flask to develop web applications, which can be useful when building the front-end of an RCTS, including payment processing and customer dashboards.

2 Python for Data Analysis

- This book is a great resource for understanding how to use Python's powerful data analysis libraries like Pandas, which would be essential for tracking and analyzing revenue data.

3.Django for Beginners: Build websites with Python and Django

- A perfect guide for learning Django, which can be used to build robust and secure back-end systems for an RCTS.

APPENDIX

(Coding)

```
def add_revenue(revenues):
    while True:
        try:
            date = input("Enter date (YYYY-MM-DD): ")
            amount = float(input("Enter amount: "))
            if date in revenues:
                revenues[date] += amount
            else:
                revenues[date] = amount
            print("Revenue entry added successfully.")
            break
        except ValueError:
            print("Invalid input. Please enter a valid amount.")

# Function to display all revenue entries
def display_revenues(revenues):
    if not revenues:
        print("No revenue entries.")
    else:
        print("Date\t\tAmount")
        for date, amount in revenues.items():
```

```
        print(f"{date}\t{amount:.2f}")



# Function to calculate total revenue
def calculate_total_revenue(revenues):
    return sum(revenues.values())

# Function to find revenue for a specific date
def find_revenue_by_date(revenues):
    date = input("Enter date (YYYY-MM-DD): ")
    if date in revenues:
        print(f"Revenue on {date}: {revenues[date]:.2f}")
    else:
        print(f"No revenue found for {date}")

# Function to export revenues to a file
def export_revenues(revenues):
    filename = input("Enter filename to export revenues: ")
    try:
        with open(filename, "w") as file:
            for date, amount in revenues.items():
                file.write(f"{date},{amount:.2f}\n")
        print("Revenues exported successfully.")
    except IOError:
        print("Error occurred while exporting revenues.")

# Function to import revenues from a file
def import_revenues(revenues):
```

```
    filename = input("Enter filename to import revenues: ")
    try:
        with open(filename, "r") as file:
            for line in file:
                date, amount = line.strip().split(",")
                revenues[date] = float(amount)
        print("Revenues imported successfully.")
    except (IOError, ValueError):
        print("Error occurred while importing revenues.")

# Main function
def main():
    revenues = {}  # Dictionary to store revenues
    while True:
        print("\nOptions:")
        print("1. Add Revenue")
        print("2. Display Revenues")
        print("3. Calculate Total Revenue")
        print("4. Find Revenue by Date")
        print("5. Export Revenues to File")
        print("6. Import Revenues from File")
        print("7. Exit")

        choice = input("Enter your choice: ")
        if choice == "1":
            add_revenue(revenues)
        elif choice == "2":
            display_revenues(revenues)
        elif choice == "3":
```

```
        print("Total Revenue:", calculate_total_revenue(revenues))
    elif choice == "4":
        find_revenue_by_date(revenues)
    elif choice == "5":
        export_revenues(revenues)
    elif choice == "6":
        import_revenues(revenues)
    elif choice == "7":
        print("Exiting program.")
        break
    else:
        print("Invalid choice. Please try again.")

# Run the program
main()
```

output

```
Options:
1. Add Revenue
2. Display Revenues
3. Calculate Total Revenue
4. Find Revenue by Date
5. Export Revenues to File
6. Import Revenues from File
7. Exit
Enter your choice:
No revenue entries.
```