Daly College,Indore

<u>2023-2024</u>

DataSense

SUBMITTED TO: -
Mr. Rajesh Nandwal

SUBMITTED BY: -
Anupam Kanoongo

INDEX

CERTIFICATE

This is to certify that this project report entitled

DataSense

has been prepared by

Anupam Kanoongo

Student of class XII. The system has been approved by the Department of Computer science, **Daly College, Indore** (M.P.) and the work has been done under my guidance. This work is up to the mark of satisfaction. We wish him success in every aspect of life. He has performed this project on his own. He has also put in sufficient periods for completion. This project has been completed as per rules of Central Board of Secondary Examination Syllabus and can be considered as the fulfillment of the A.I.S.S.C.E. Examination.

Date:

Mr. Rajesh Nandwal
Head, Department of Computer Science,
Daly College, Indore
Indore.

ACKNOWLEDGMENT

I feel immense pleasure and deep feeling of gratitude towards ***Mr. Rajesh Nandwal*** **(H.O.D [Comp. Sc.])** of **Daly College, Indore** for his skillful guidance constructive and valuable suggestion and encouraging co-operation for my project, which not merely helped but enable me to give effort towards this project.

I am also thankful to D***r Gunmeet Bindra***, **Principal**, **Daly College, Indore** for his encouragement and valuable suggestions given to me from time to time, I also extend my thanks to all my colleagues and friends for their valuable assistance and kind co-operation during the course of this investigation.

Date:

Place: Daly College

Anupam Kanoongo

INTRODUCTION

The purpose of this project is to develop a Python app, named Data Sense, that can analyze the sales data of the food industry stored in a Remote Database. The app provides both visual and numeric analysis of the data and offers a GUI interface for users to access and analyze the data directly from the Remote Database in fraction of time.

The app utilizes the following libraries and APIs:

- **Mysql.connector** to build and manage the database.
- **NumPy** for performing numerical analysis on the data.
- **Matplotlib** for creating graphs and visualizations of the data.
- **Python Image Library (PIL)**, and **Input/Output(io)** library for GUI enhancement and data handling.
- **SecuriPy**, a library developed by the app's developer, for security-related functionalities.
- **Datetime**, used to manage records with proper date and time values.

DATA DICTIONARY

The following functions have been used in the project:

1. **The SQL - Python Interface :-**
   a. **connect_to_database():**
   Establish a connection to the database.

   b. **execute_query():**
   Execute a SQL query on the connected database.

   c. **close_connection():**
   Close the connection to the database.

   d. **custom_query_save_excel():**
   Execute a custom SQL query and save the result to an Excel file.

   e. **custom_query_save_csv():**
   Execute a custom SQL query and save the result to a CSV file.

   f. **save_excel():**
   Save data from the database to an Excel file.

   g. **save_csv():**
   Save data from the database to a CSV file.

2. **Numeric Queries :-**
   a. **get_product_categories():**
   Retrieves distinct product categories from the "Products" table.

   b. **get_customer_orders(customer_id):**
   Retrieves order details (OrderID, OrderDate, TotalAmount) for a specific customer.

c. **get_high_priced_products(min_price):**
   Retrieves products with prices greater than or equal to a specified minimum price.

d. **get_order_count_by_customer():**
   Retrieves the count of orders for each customer.

e. **get_orders_in_date_range(start_date, end_date):**
   Retrieves orders within a specified date range.

f. **get_total_revenue_by_category():**
   Retrieves the total revenue for each product category.

g. **get_customer_total_spent():**
   Retrieves the total amount spent by each customer.

h. **get_average_product_price_by_category():**
   Retrieves the average product price for each category.

i. **get_order_details(order_id):**
   Retrieves details of products in a specific order.

j. **get_products_in_category_cat(category):**
   Retrieves products and prices for a specific product category.

k. **get_products_in_category():**
   Retrieves products and prices for all product categories.

l. **get_customer_details(custid):**
   Retrieves details for a specific customer.

m. **get_customers_with_highest_spending():**
   Retrieves the top 5 customers with the highest total spending.

n. **get_orders_by_date_and_category(date, category):**
   Retrieves order details for a specific date and product category.

o. **get_orders_by_date(date):**
   Retrieves order details for a specific date.

p. **get_total_revenue_by_product():**
   Retrieves the total revenue for each product, showing the top 5.

3. **Visual Queries :-**

   a. **product_categories():**
      Retrieves a list of distinct product categories from the "Products" table.

   b. **order_dates():**
      Retrieves a list of distinct order dates from the "Orders" table.

   c. **products():**
      Retrieves a list of product names from the "Products" table.

   d. **prod_price():**
      Retrieves a list of product prices from the "Products" table.

   e. **customers():**
      Retrieves a list of customer names (concatenation of first and last names) from the "Customers" table.

4. **Graphing Tools :-**

   a. **bar_chart(x_values, y_values, x_axis_label, y_axis_label, title, orientation='vertical'):**
      Generates a bar chart with customizable x-axis values, y-axis values, axis labels, and title, allowing for both vertical and horizontal orientations.

b. **pie_chart(labels, sizes, title):**
   Creates a pie chart with specified labels and sizes, displaying the percentages and random colors for each segment.

5. **Windows Defined :-**

   a. **home_window():**
   Defines a Tkinter-based GUI function for a home window with buttons for data manipulation, analysis, export, account management, and logout.

   b. **signup_window():**
   Creates a Tkinter-based GUI function for a signup window, enabling users to register with a username and password. The function includes validation checks, encryption, and options to navigate back to the login page or exit the application.

   c. **login_window():**
   Creates a Tkinter-based GUI function for a login window, allowing users to enter their username and password for authentication. The function includes login validation, error handling, and options to navigate to the home page, sign up for a new account, or exit the application.

   d. **add_window():**
   Creates a Tkinter-based GUI function for a window to add/insert data, providing options to add new customers, products, or orders. The function includes data submission forms with input fields, validation checks, and options to navigate back to the home page or exit the application.

   e. **update_window():**
   Creates a Tkinter-based GUI function for a window to update data, offering options to update customer details or product details. The function includes data submission forms with input

fields, validation checks, and options to navigate back to the home page or exit the application.

f. **accounts_window():**
Creates a Tkinter-based GUI function for an accounts management window, providing options to change the password or delete the account. The function includes input fields, validation checks, and options to navigate back to the home page or exit the application.

g. **delete_window():**
Creates a Tkinter-based GUI function for an accounts management window, providing options to change the password or delete the account. The function includes input fields, validation checks, and options to navigate back to the home page or exit the application.

h. **welcome_window():**
Creates a Tkinter-based GUI function for a welcome page, providing information about the DataSense app and options to create an account, log in, or exit the application. The function includes text, buttons, and graphical elements to enhance the user interface.

i. **numeric_window():**
Creates a Tkinter-based GUI function for a numeric analysis page, featuring buttons for actions such as retrieving customer details, order information, product categories, and more. The function incorporates user input dialogs and result displays for enhanced user interaction and data presentation.

j. **export_window():**
   Creates a Tkinter-based GUI for exporting data, providing options to export tables as CSV, spreadsheet, or through custom queries (CSV and spreadsheet). The interface includes buttons for each export option, along with the ability to exit, return home, and refresh. The design features an appealing background image for an enhanced user experience.

k. **about_window():**
   Displays an about page using Tkinter, providing information about the DataSense project. The page includes details about the project's purpose, key features (data analysis, database management, and security), the developer, and a conclusion. Users can navigate back to the home page or exit the application through corresponding buttons. The page layout is designed with an appealing background image for an enhanced visual experience.

SYSTEM REQUIREMENTS

Software

- *Microsoft Windows 2007 or more*
- *Python 3.10 or more*
- *MySQL v8 and further*
- *Editor like IDLE, Sublime, PyCharm*

Hardware

- *Min i3 processor*
- *Min 4 GB RAM*
- *Min 10 GB HDD*

ADVANTAGES AND LIMITATIONS

Advantages:

1. Graphical User Interface (GUI): The code utilizes Tkinter to create a graphical user interface, making it user-friendly and accessible.
2. Modular Design: The code is organized into functions, promoting modular design. Each function handles specific tasks, enhancing code readability and maintainability.
3. Database Interaction: The application interacts with a SQL database, performing various operations such as extracting data, executing custom queries, and exporting data in different formats (CSV, spreadsheet).
4. Data Analysis and Visualization: The inclusion of functions for numeric and visual analysis (e.g., total sales, percentage of total sales, sales by product) demonstrates a capability for data analysis and visualization.
5. Security Measures: The mention of "SecuriPy" suggests the inclusion of security measures, addressing data security concerns. However, the specific security measures are not detailed in the provided code.
6. Use of External Libraries: The code uses external libraries like PIL (Pillow) for image processing, enhancing its functionality.
7. Error Handling: The code includes basic error handling, such as confirmation dialogs for quitting the application.

Limitations:

1. Limited Error Handling: While some basic error handling is present, the code might benefit from more comprehensive error handling to handle various scenarios and enhance robustness.
2. Code Duplication: There is some code duplication in GUI-related operations, such as window setup. Refactoring to reduce redundancy could improve code maintainability.
3. Security Implementation Not Detailed: Although security measures are mentioned, the specific implementation details and the effectiveness of "SecuriPy" are not provided in the code.
4. User Input Handling: The code uses simpledialog for user inputs, which may not be the most user-friendly approach. Additionally, it lacks input validation for ensuring the correctness of user inputs.
5. Documentation: The code lacks inline comments and documentation, making it challenging for others (or even the original developer) to understand and maintain the code in the long run.
6. Visual Design: While the code includes background images to enhance the visual appeal, the overall visual design could be improved for a more polished appearance.
7. Limited Testing: The code lacks explicit testing, making it difficult to assess its reliability and robustness under different scenarios.
8. Limited Scalability: As the code grows, the lack of a clear structure for handling larger-scale applications may hinder its scalability.

FUTURE ENHANCEMENT

User Authentication and Authorization:
Implement a secure user authentication system to ensure that only authorized users can access the application.
Integrate role-based access control (RBAC) to manage different levels of user privileges.

Improved Graphical User Interface (GUI):
Enhance the overall visual design and layout of the application to provide a more modern and user-friendly experience.
Implement responsive design principles to optimize the application for different screen sizes.

Advanced Data Analysis Features:
Introduce more advanced data analysis features, such as predictive analytics, trend forecasting, and machine learning integration.
Incorporate a wider range of data visualization options and customization settings for graphs and charts.

Enhanced Data Export Options:
Expand data export capabilities to support a variety of formats beyond CSV and Excel, such as JSON, XML, or database-specific formats.
Provide options for selective data export based on user-defined criteria.

Database Connection Management:
Implement a feature to manage and switch between multiple database connections seamlessly.
Allow users to define and save different database connection profiles for quick access.

Real-Time Data Updates:
Enable real-time data updates to reflect changes in the database dynamically.
Implement automatic refresh intervals or push notifications to keep users informed of the latest data.

CONCLUSION

In conclusion, developing this project proved to be a valuable learning experience, offering clarity on complex topics within the syllabus. It not only introduced a novel programming approach but also fostered the exploration of diverse logical solutions to common challenges.

The resulting software significantly streamlines computer management, enhancing efficiency and reducing the time and resources traditionally invested in manual tasks. By optimizing processes, it stands to save both time and money—two invaluable resources in the current era. Upon implementation, it holds the potential to revolutionize school database management, offering a more effective and economical solution.

BIBLIOGRAPHY

S.No	Name	Author
1.	Informatics Practices XI	Sumita Arora
2.	Informatics Practices XII	Sumita Arora
3.	Informatics Practices XII	C.B.S.E