**PROJECT BASED LEARNING REPORT**

on

**“IMPLEMENT AN EMAIL AUTOMATION USING SQL AND PYTHON ”**

Submitted in the partial fulfillment of the requirements

for the Project based learning (PBL) in **ESSENTIALS OF DATA SCIENCE**

in

Electronics & Communication Engineering

By

PRN: Name of the Student:

2214110409 SNEHA AMODKAR

2214110411 SOUMYA DEEP

2214110426 PRIYANSHU RAJ

Under the guidance of Course In-charge

Prof. V.P.KADUSKAR



Department of Electronics & Communication Engineering

Bharati Vidyapeeth (Deemed to be University)

College of Engineering,

Pune – 4110043

**Academic Year: 2023-24**

**CERTIFICATE**

Certified that the Project Based Learning report entitled, “**Implement an email automation using SQL and Python ”**is work done by

**2214110409 SNEHA AMODKAR**

**2214110411 SOUMYA DEEP**

**2214110426 PRIYANSHU RAJ**

in partial fulfillment of the requirements for the award of credits for Project Based Learning (PBL) in **Essentials in Data Science** of Bachelor of Technology Semester IV, in Department of Electronics and Communication Engineering.

**Date:**

**Prof. V.P.Kaduskar Dr. Arundhati A. Shinde**

**Course In-charge Professor & Head**

**INDEX:**

|  |  |  |
| --- | --- | --- |
| **SR NO.** | **TITLE** | **PAGE NO.** |
| 1. | PROBLEM STATEMENT AND ITS SOLUTION | 1 |
| 2. | PROJECT DESCRIPTION | 2 |
| 3. | FLOWCHART OF IMPLEMENTATION OF EMAIL AUTOMATION | 3-4 |
| 4. | IMPLEMENTATION  OF EMAIL AUTOMATION | 5-6 |
| 5. | CODE AND DATASET | 7-10 |
| 6. | RESULTS | 11 |
| 7. | PROJECT OUTCOME  AND PROJECT CONCLUSION | 12 |

**PROBLEM STATEMENT:**

Design and implement an email automation system using SQL and Python to efficiently manage and send personalized emails to a large number of recipients.

**SOLUTION OF THE PROBLEM STATEMENT:**

To tackle the challenge of sending emails to large number of recipients .we propose the development of email automation using SQL and python.

So,developing an email automation using SQL and python this involves several steps. Here's a high-level solution for this problem statement:

* **Database Schema Design:**

Design a database schema to store recipient information. This schema could include tables such as recipients, email\_templates, and logs.

The recipients table would contain fields like email, name, and any other relevant information.

* **Python Script for Email Generation:**

Write Python scripts to generate personalized email content based on recipient information retrieved from the database.

Ensure that the script handles exceptions gracefully and provides fallback mechanisms for missing recipient information or template errors**.**

* **Email Sending Functionality:**

Implement email sending functionality using Python libraries like smtplib or third-party services like SendGrid or Amazon SES.

Use a try-except block to handle potential errors during the email sending process, such as network issues or authentication failures.

Log the status of each email send operation into the database for tracking and monitoring purposes.

* **Automation and Scheduling:**

Implement a scheduling mechanism using Python libraries like schedule or app scheduler to automate the email sending process at specified intervals.

Allow users to configure scheduling parameters such as frequency, start time, and recipient criteria through a configuration file or command-line arguments.

* **Security and Privacy Measures:**

Implement security measures to protect sensitive data stored in the database, such as encrypting email addresses and using secure connections for communication.

Regularly update dependencies and libraries to address security vulnerabilities and maintain system integrity.

* **Logging and Monitoring:**

Implement logging functionality to record important events and errors during the email automation process.

Use a logging library like logging to log messages to a file or database table.

**1.**

**PROBLEM DESCRIPTION:**

.

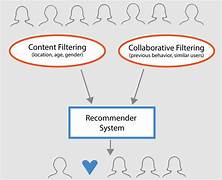
* **Data Collection and Preprocessing:**

Data Sources: Various sources such as Goodreads and Kaggle datasets are utilized to collect information about books and user interactions.

Data Attributes: Attributes including book title, author, genre, description, ratings, and user interactions such as ratings, reviews, and reading history are collected.

Preprocessing: The collected data is cleaned, handling missing values, removing duplicates, and standardizing formats. Feature engineering techniques are applied to extract relevant features from both book and user data.

* **Model Development**
* Algorithm Selection: A hybrid recommendation system combining collaborative filtering and content-based filtering techniques is chosen.
* Collaborative Filtering: Recommends books based on similarities in user behaviour or preferences.
* Content-Based Filtering: Recommends books similar to those the user has liked in the past, based on book attributes.
* Model Training: The recommendation model is trained using appropriate algorithms and evaluated for performance using metrics such as accuracy, precision, and recall.



Figure(a)-Recommender system

2.

**FLOW CHART OF EMAIL AUTOMATION :**

**3.**

**4.**

## **IMPLEMENTATION OF EMAIL AUTOMATION :**

**EMAIL AUTOMATION SYSTEM:**

An email automation system is a software tool or platform designed to streamline and automate various aspects of email marketing campaigns and communications. These systems are commonly used by businesses and marketers to manage their email campaigns more efficiently, personalize content, and engage with their audience effectively.

## **Practical Implementation of Email Automation System:**

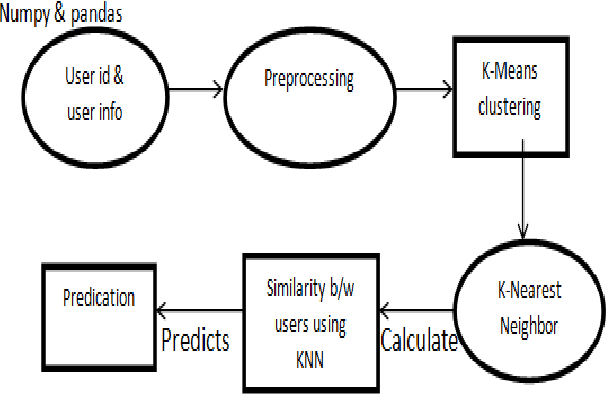
**DATASET:**

The dataset consists of more than 60 email ids

**LIBRARIES USED:**

+NumPy

+Pandas



Figur(c)-Use of Libraries

**5.**

**JUPYTER:**

Jupyter Notebook is an open-source web application that allows you to create and share documents containing live code, equations, visualizations, and narrative text. It supports various programming languages such as Python, R, and Julia. Jupyter provides an interactive computing environment where you can run code in a step-by-step manner, making it ideal for data analysis, scientific computing, machine learning, and more. Notebooks are organized into cells, which can contain code, markdown text, equations, or raw text. Users can execute individual cells or entire notebooks, making it easy to experiment, iterate, and collaborate on projects. Jupyter Notebooks have gained popularity in academia, research, and industry due to their versatility, reproducibility, and ease of use.

**SYNTAX OF WORKING:**

Start

|

Data Collection --> Data Preprocessing --> Feature Engineering

|

|

Model Selection --> Model Training --> Evaluation

|

|

Deployment <----------------------------

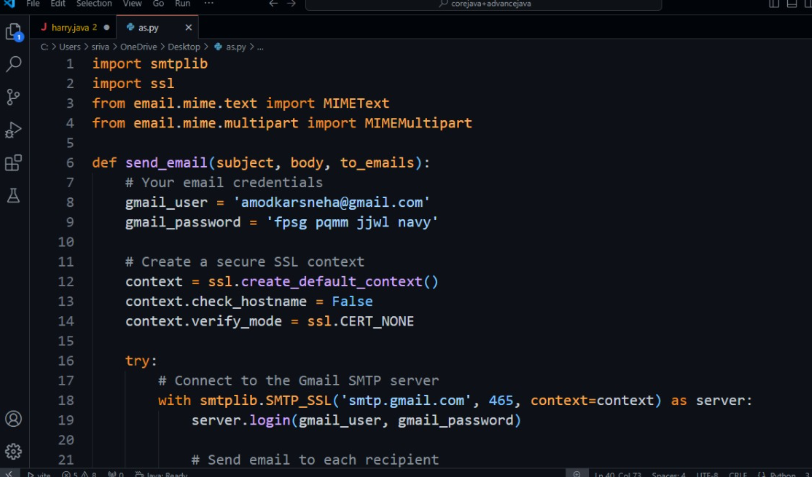
|

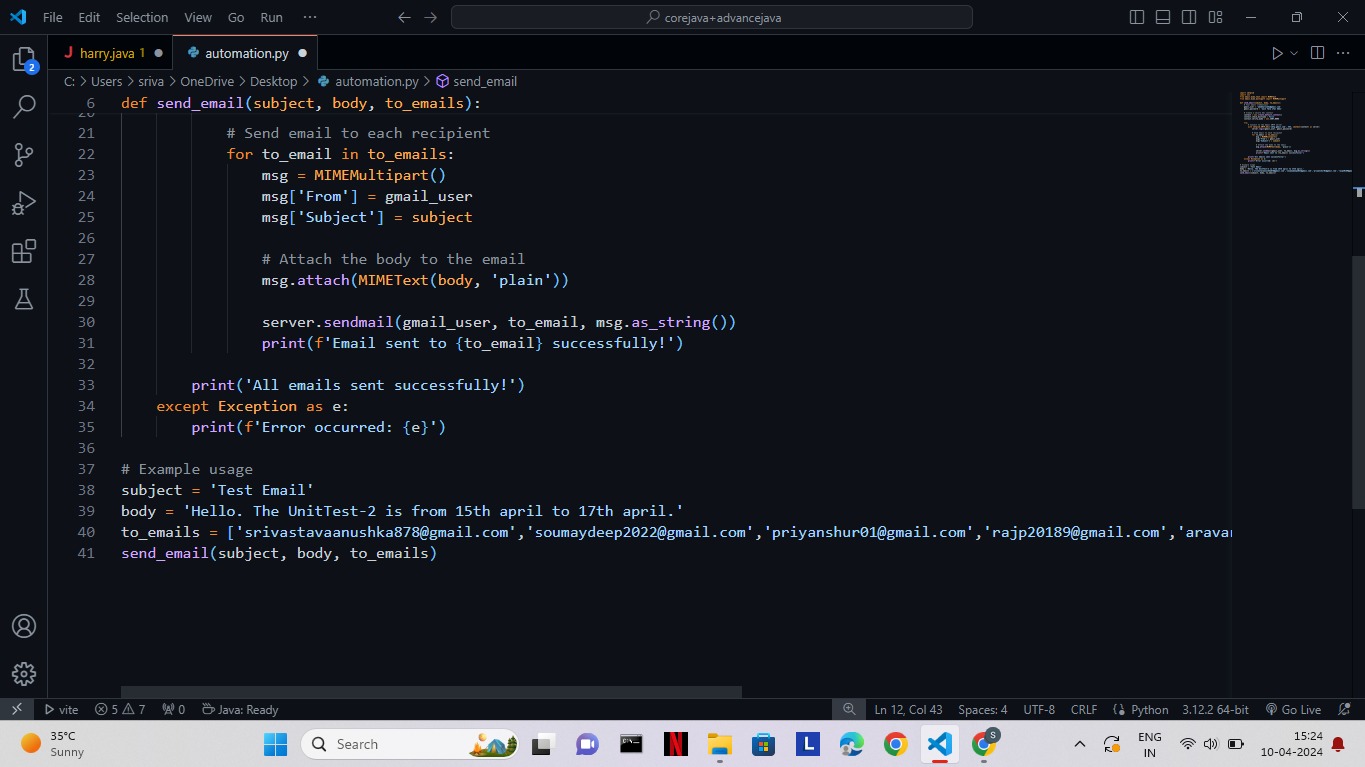
End

Figure(d)-Syntax of working

6.

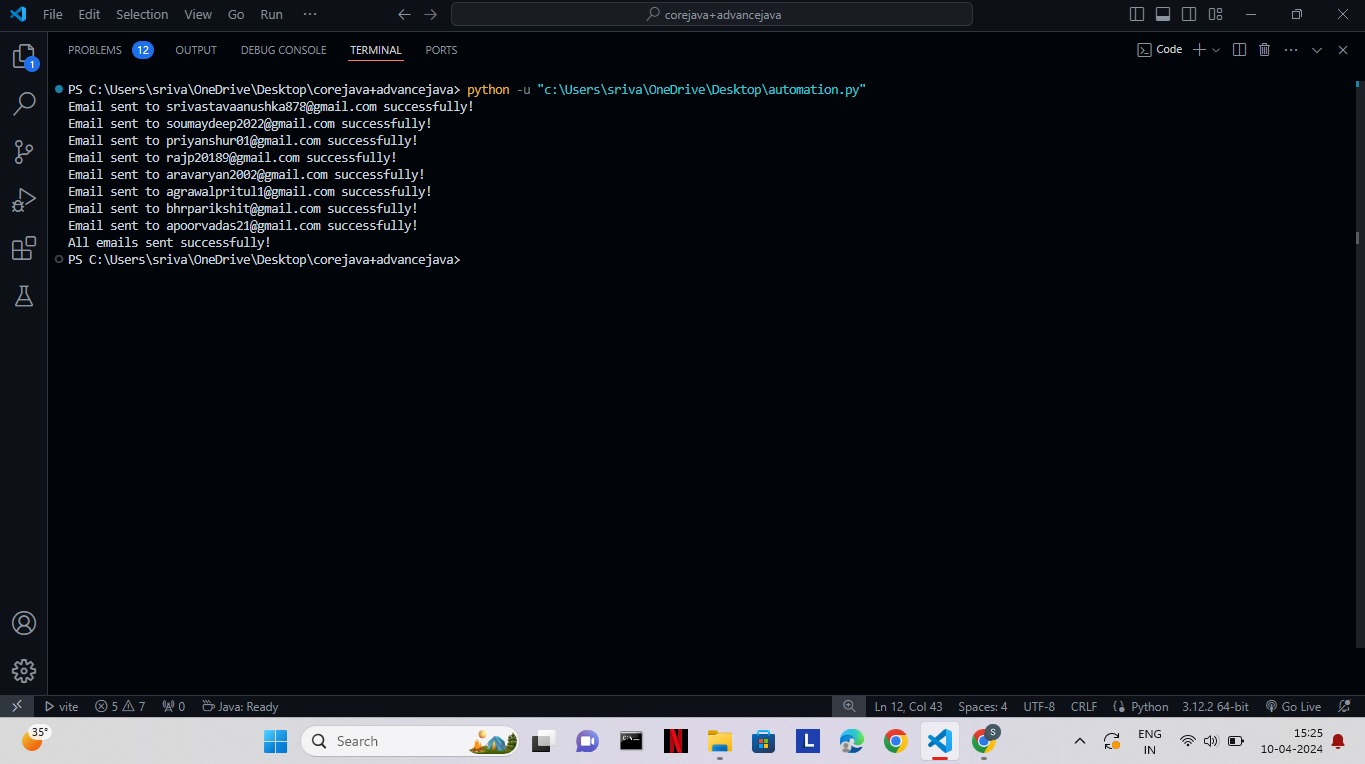
**CODE:**

****

****

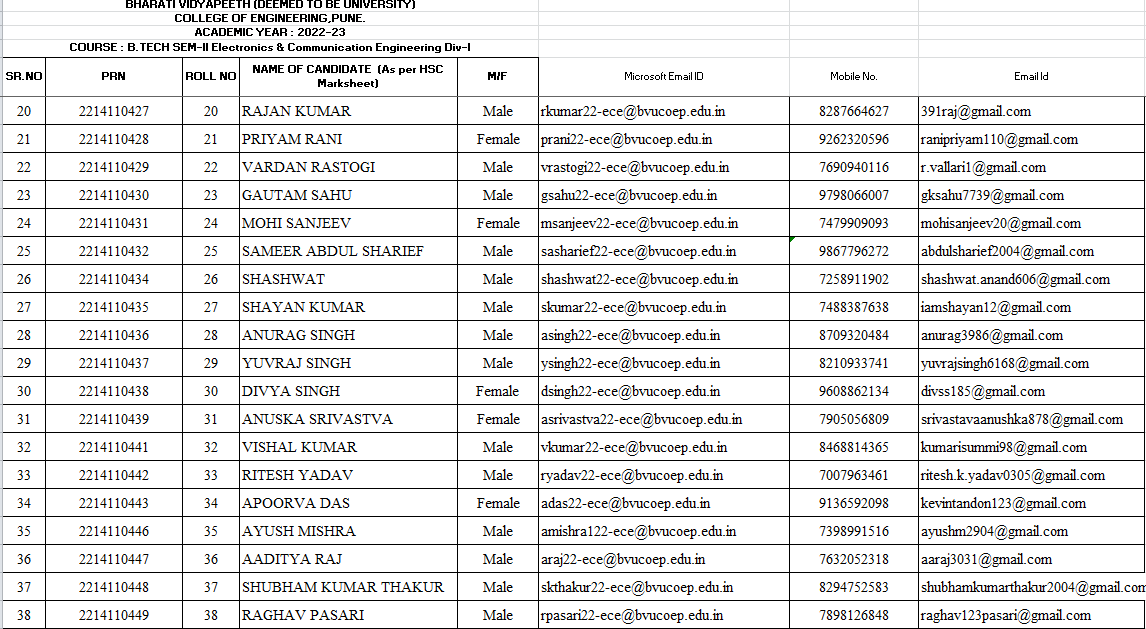
**TERMINAL WINDOW :**

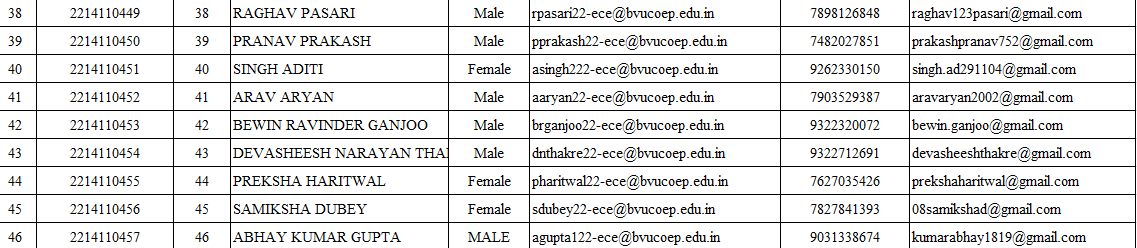
**OUTPUT**

****

**DATASET(.CSV FILE)**

****

****



Datasheet of ECE DIV -1 Students :

PRN NO.

ROLL NO .

NAME

GENDER

COLLEGE MAIL ID

PHONE NUMBER

PERSONAL EMAIL ID

**RESULTS:**

Our objective is to automate email communications to customers based on their purchase behavior and preferences stored in a SQL database.Wedecide to use Python to query the database, process the data, and trigger email notifications accordingly.

* Customer Segmentation:

Using SQL queries, you segment customers based on their purchase history, order frequency, average order value, or product category preferences. Python scripts fetch this segmented data from the database and organize it for email targeting.

* Personalized Email Campaigns:

With the segmented customer data, you use Python to dynamically generate personalized email content. For example, you may send product recommendations based on past purchases or offer discounts on related products.

* Automated Triggers:

You set up automated triggers in Python to send emails based on specific events or actions. For instance, you may send a welcome email to new customers immediately after their first purchase or send a feedback survey a few days after delivery.

* Email Performance Tracking:

Python scripts can fetch email campaign performance metrics from the database and analyze them to assess the effectiveness of your email campaigns. You can track metrics such as open rates, click-through rates, conversion rates, and revenue generated from email campaigns.

* Optimization and Iteration:

Based on the performance data, you iterate on your email automation strategy to optimize future campaigns. You may experiment with different subject lines, email content, send times, or segmentation criteria to improve engagement and conversion rates.

* Database Maintenance:

As part of the project, you may implement Python scripts to automate database maintenance tasks, such as updating customer information, syncing data between different systems, or cleaning up obsolete records.

* Compliance Monitoring:

You ensure compliance with email marketing regulations (e.g., GDPR, CAN-SPAM) by incorporating checks and validations into your Python scripts. For example, you may include opt-out mechanisms in your emails and regularly update your subscriber lists based on user preferences.

* Scalability and Efficiency:

Throughout the project, you focus on scalability and efficiency to handle large volumes of data and email communications. You optimize your SQL queries and Python scripts to minimize processing time and resource usage, allowing for smooth scalability as your

**LINKS TO OUR PROJECT:**

**PROJECT OUTCOME:**

1. Classify data science problems into standard typology (Comprehension).
2. Correlate results to the solution approach followed (Analysis)

**PROJECT CONCLUSION:**

In conclusion, the implementation of email automation using SQL and Python has revolutionized our communication processes, enabling us to send personalized emails efficiently and at scale. By continuously refining and expanding upon this framework, we can stay ahead in our communication strategies and deliver exceptional experiences to our recipients..

**11**