Міністерство освіти і науки України

Національний університет «Чернігівська політехніка»

Кафедра інформаційних технологій та програмної інженерії

Курсовий проект з дисципліни

«Комп’ютерні мережі»

Розсилка рекламних листів на електрону пошту

Виконавець:

студент гр. КІт-221

Коржев Олександр Андрійович

*(підпис)*

Керівник:

|  |  |  |
| --- | --- | --- |
|  |  | *ст. викладач* |
| **–––––––––––––––––––––––––––––** |  | **–––––––––––––––––––––––––––––––––––––––––––––––––––––––––––––––––––––––––––––––––––––––––––** |
| *(посада)* |  | *(науковий ступінь, вчене звання)* |
|  |  |  |

Риндич Євген Володимирович

*(підпис)*

Чернігів, 2022 рік

# Технічне завдання

На виконання курсового проекту з дисципліни «Комп’ютерні мережі»

студента Коржев О.А. , група КІт-221

**Тема роботи:** «Bulk Email Sender»

**Завдання на роботу:**

As the name of the project suggests it's working. It is a gem for companies that need to send thousands of emails every day to promote their products. You can build this with the help of an email package available in python.

Результатом курсового проекту є працездатний додаток та звіт обсягом приблизно 20 сторінок друкованого тексту оформленого відповідно до стандартів кафедри.

**Очікувані технічні та експлуатаційні результати роботи:**

В результаті виконання роботи має бути створена програма мовою Python для розсилання листів на адреси користувачів які занесені до бази даних

Програма завантажує список користувачів з БД, далі можна налаштувати повідомлення, також можна обрати готові варіанти з БД і почати розсилку на всі скриньки.

Система має:

* Підключатись до бази даних
* Завантажувати дані з бази даних
* Розсилати лист на всі завантажені скриньки

Інтерфейс користувача має:

* + Поля для вводу даних
  + Список шаблонів повідомлень
  + Кнопку для запуску розсилки

**Планова трудомісткість роботи:**1 кредит (30годин).

**Обсяг текстової документації:**

Пояснювальна записка до проекту об'ємом приблизно 20 сторінок друкованого тексту формату А4. Обсяги текстової інформації можуть бути скориговані в процесі роботи за погодженням з керівником.

Документація оформляється згідно СОККР-ІКС-2001

**Плановий термін захисту роботи:** 13-14 тижні семестру.

Завдання підготував:

керівник Риндич Євген Володимирович

(підпис)

«\_\_\_» \_\_\_\_\_\_\_\_\_ 2022 р.

Завдання одержав:

студент Коржев Олександр Андрійович

(підпис)

«\_\_\_» \_\_\_\_\_\_\_\_\_ 2022 р.

# Реферат

Курсовий проект 21с., 12 рис., 9 лістингів, 6 джерел.

У роботі має бути описаний процес і результати проектування та розробки (реалізації) системи «Масова розсилка електронної пошти».

Мета курсового проєкту полягає в поглибленому вивченні навчального матеріалу з курсу «Комп’ютерні мережі». Результат проєктування та розробки системи наданий у вигляді працюючого python-додатка, пояснювальної записки до курсового проєкту та лістингів програмного коду, які наведені в записці.

Дана інформаційна система може бути розроблена в будь-якому середовищі програмування, з використанням мови програмування Python. При розробці використовуватиметься бібліотеки візуальних компонентів PyQt, стандартна бібліотека для роботи з поштою smtplib та бібліотека для підкючення до бази даних mysql connector

Робота інформаційно-довідкової системи можлива в операційних системах з встановленим інтерпритатором Python

Подальший розвиток системи можливий у наступних напрямках:

* оптимізація роботи додатку;
* оптимізація коду прогами для збільшення продуктивності виконання та забезпечення безпеки;
* покращення відуалької складової додатку: збільшення кількості анімованих елементів, підвищення якості відображення;
* реалізація додаткових налауштвань роботи системи;

Ключові слова: ПРОГРАМУВАННЯ, Python, Комп’ютерні мережі, Бази даних, електрона пошта, розсилка

**Зміст**

[Технічне завдання 2](#_Toc122980770)

[Реферат 4](#_Toc122980771)

[Вступ 6](#_Toc122980772)

[1 Проектування системи 7](#_Toc122980773)

[1.1 Опис предметної області 7](#_Toc122980774)

[1.2 Аналіз вирішуваної задачі 7](#_Toc122980775)

[2 Аналіз поведінки об’ктів системи 9](#_Toc122980776)

[2.1 Пошта для розсилки 9](#_Toc122980777)

[2.2 Клас розсилки 9](#_Toc122980778)

[2.3 База даних 9](#_Toc122980779)

[2.4 Підключення до бази даних 10](#_Toc122980780)

[2.5 Головний клас програми 11](#_Toc122980781)

[3 Реалізація системи 12](#_Toc122980782)

[3.1 Створення класу Spammer 12](#_Toc122980783)

[3.2 Створення бази даних 13](#_Toc122980784)

[3.3 Створення головного класу 14](#_Toc122980785)

[3.3.1 Створення інтерфейсу 14](#_Toc122980786)

[3.3.2 Підключення до бази даних 15](#_Toc122980787)

[3.3.3 Початок розсилки 15](#_Toc122980788)

[3.4 Запуск програми 16](#_Toc122980789)

[4 Тестування системи 19](#_Toc122980790)

[Порівняння з аналогами 20](#_Toc122980791)

[ВИСНОВКИ 23](#_Toc122980792)

[ВИКОРИСТАНІ ДЖЕРЕЛА 24](#_Toc122980793)

# Вступ

Великі компанії часто роблять розсилку електроних листів для свої клієнтів які на неї підписались. Це допомає рекламувати новий продукт та дозволяє клієнтам отримувати інофрмацвю про новий товар. Тому дуже зручно мати додаток який би автоматично виконував бі розсилку на адреси з бази даних сервісу.

Мета дослідження полягає в розробці python-додатка з графічним інтерфейсом, який виконує розсилку.

# 1 Проектування системи

# 1.1 Опис предметної області

Після завантаження додаток підключається до бази даних, отримуючи список підписаних користувачів і шаблони листів. Далі можна ввести дані від пошти з якої буде вестись розсилка та описати лист. Після натискання на кнопку має виконуватись розсилка.

# 1.2 Аналіз вирішуваної задачі

Основна задача проекта полягає в реалізації програми, що може виконувати розсилку на електроні скриньки.

Для розв’язання задачі необхідно використати готові бібліотеки, що можуть дозволити реалізувати необхідний функціонал. Також необхідно розділити проект на клас який безпосередньо виконує розсилку та головний клас з реалізацією. Таким чином функціонал можна буде перевикористовувати.

Для взаємодії з застосунком необхідно також створити графічний інтерфейс.

Аналізуючи предметну область можна виділити наступні складові системи:

* Бібліотека для розсилки
* Клас розсилки
* Поштова скринька для розсилки
* Графічний інтерфейс
* Налаштування розсилки
* Конектор до бази даних

Схематичне зображення структури проекта показано на рисунку 1.1

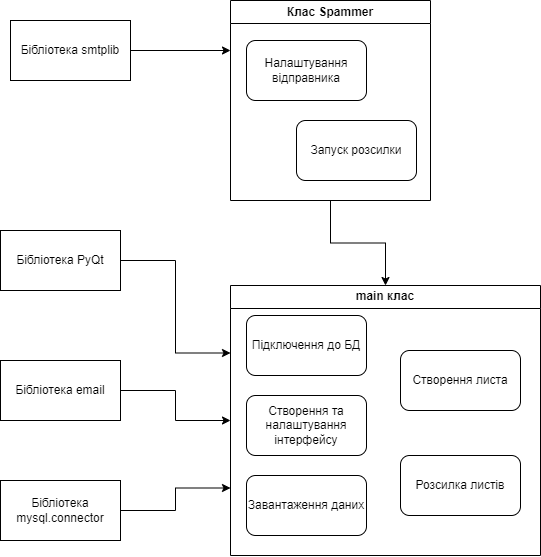


Рисунок 1.1 – Структура проекта

**1.3 Вибір бібліотек та мови програмування**

Для створення візьмемо мову програмування Python. Це високорівнева об’єкто-орієнтована мова програмування, що має великий асортимент бібліотек як для графічного інтерфейсу так і для роботи з електронною поштою.

Серед бібліотек оберемо такі як:

* smtplib – стандартна бібліотека мови Python що дозволяє працювати з електронною поштою.
* PyQt – фреймворк для розробки графічного інтерфейсу
* EmailMessage – бібліотека для створення електронних листів
* MySql – бібліотека що дозволяє працювати з базою даних MySQL

# 2 Аналіз поведінки об’ктів системи

# 2.1 Пошта для розсилки

Для розсилання листів створемо поштову скриньку на сервісі Gmail. Саме з неї буде виконуватись розсилка на пошту підписників

# 2.2 Клас розсилки

Безпосередньо для розсилки створемо клас який можна буде використовувати у майбутньому. Схему класа наведена на рисунку 2.1

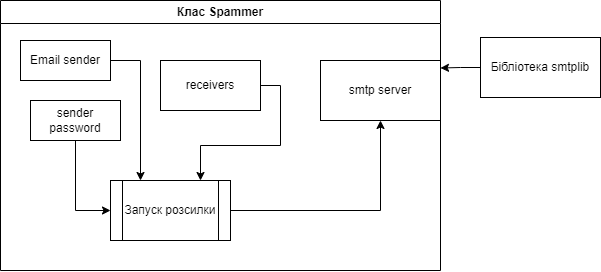


Рисунок 2.1 – Схема сутності для розсилки

# 2.3 База даних

Дані будемо зберігати в базі даних з двома таблицями.

Таблиця Subscribers відповідатиме за користувачів які підписались на розсилку (через сайту або по іншому)

Таблиця Messages відповідатиме за шаблони повідомлень які можна використати для створення листів для розсилки

Схематичний вигляд бази даних наведено на рисунку 2.2

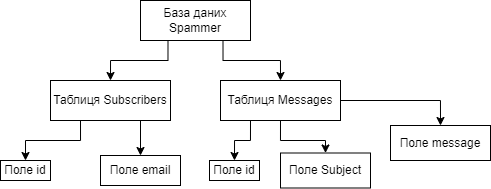


Рисунок 2.2 – Схема бази даних

# 2.4 Підключення до бази даних

Для підключення будемо використовувати бібліотеку mysql. Блок схема алгоритма підключення показана на рисунку 2.3

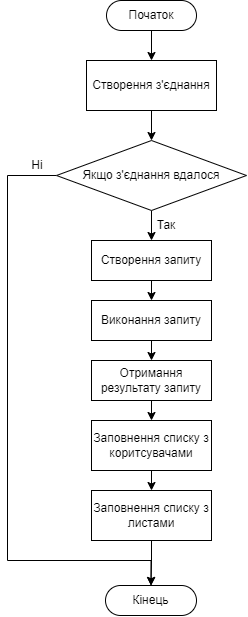


Рисунок 2.3 – Блок схема роботи з БД

# 2.5 Головний клас програми

Запуск програми буде виконуватись з класу main. Тут будуть створюватись усі необхідні об’екти, також він відповідатиме за відображення інтерфейсу. Схема діяльності наведена на рисунку 2.4

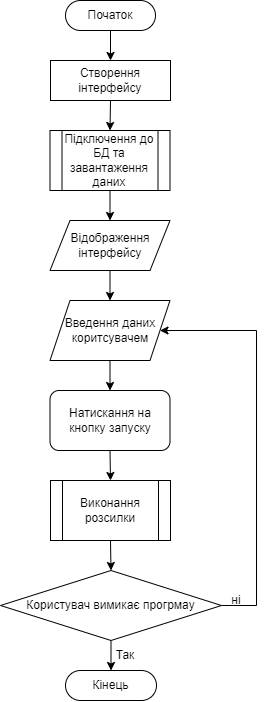


Рисунок 2.4 – Блок схема діяльності програми

# 3 Реалізація системи

# 3.1 Створення класу Spammer

Клас Spammer відповідатиме за розсилку повідомлень. Підключимо до класу бібліотеку smtplib.

*import smtplib*

В конструкторі будемо приймати пошту та пароль відправника, а також список отримувачів. Код конструктора наведено у Лістинзі 1

Лістинг 1 – Код конструктора класу Spammer

*def \_\_init\_\_(self, email\_sender, sender\_password, receivers):*

*self.email\_sender = email\_sender*

*self.sender\_password = sender\_password*

*self.receivers = receivers*

*self.smtp\_server = smtplib.SMTP('smtp.gmail.com', 587)*

Для запуску розсилки створемо метод run який параметром прийматиме повідомлення. Код методу наведено у Лістинзі 2

Лістинг 2 – Код метода run

*def run(self, message):*

*self.smtp\_server.starttls()*

*self.smtp\_server.login(self.email\_sender, self.sender\_password)*

*self.smtp\_server.sendmail(self.email\_sender, self.receivers, message.as\_string())*

Код усього класу наведено у Лістинзі 3

Лістинг 3 – Код класу Spammer

*import smtplib*

*class Spammer:*

*def \_\_init\_\_(self, email\_sender, sender\_password, receivers):*

*self.email\_sender = email\_sender*

*self.sender\_password = sender\_password*

*self.receivers = receivers*

*self.smtp\_server = smtplib.SMTP('smtp.gmail.com', 587)*

*def run(self, message):*

*self.smtp\_server.starttls()*

*self.smtp\_server.login(self.email\_sender, self.sender\_password)*

*self.smtp\_server.sendmail(self.email\_sender, self.receivers, message.as\_string())*

# 3.2 Створення бази даних

Для зберігання даних створимо базу даних. В якості БД візьмемо MySQL. Код створення таблиці Messages наведено у Лістинзі 4

Лістинг 4 – Код створення таблиці Messages

*CREATE TABLE `messages` (*

*`id` int NOT NULL AUTO\_INCREMENT,*

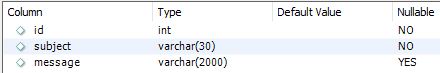
*`subject` varchar(30) NOT NULL,*

*`message` varchar(2000) DEFAULT NULL,*

*PRIMARY KEY (`id`)*

*) ENGINE=InnoDB AUTO\_INCREMENT=3 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;*

Вигляд таблиці показано на рисунку 3.1



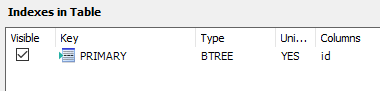


Рисунок 3.1 – Вигляд таблиці Messages

Далі створимо таблицю Users. Код таблиці показано у Лістинзі 5

Лістинг 5 – Код таблиці Users

*CREATE TABLE `subscribers` (*

*`subscriber\_id` int NOT NULL AUTO\_INCREMENT,*

*`email` varchar(45) NOT NULL,*

*PRIMARY KEY (`subscriber\_id`),*

*UNIQUE KEY `user\_id\_UNIQUE` (`subscriber\_id`)*

*) ENGINE=InnoDB AUTO\_INCREMENT=3 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci*

Вигляд таблиці показано на рисунку 3.2



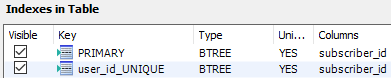


Рисунок 3.2 – Вигляд таблиці Users

# 3.3 Створення головного класу

# 3.3.1 Створення інтерфейсу

Для інтерфейсу використаємо бібліотеку PyQt6. Відображати інтерфейс будемо у класі Window який наслідуємо від QWidget. В конструкторі будемо створювати віджети та додавати їх до вікна. Приклад створення віджету навдено у Лістинзі 6

Лістинг 6 – Код створення віджету

*layout.addWidget(QLabel('Sender password'), 1, 0)*

*self.passwordEdit = QLineEdit()*

*self.passwordEdit.setEchoMode(QLineEdit.EchoMode.Password)*

*self.passwordEdit.setText('srioaofjjlrlltwe')*

*layout.addWidget(self.passwordEdit, 1, 1)*

# 3.3.2 Підключення до бази даних

Для підключення використаймо бібліотеку mysql, щоб створити підключення необхідно використати метод connect з введенням даних БД. Далі перевіряємо підключення, створюємо запит і отримуємо дані. Код реалізації наведено у Лістинзі 7

Лістинг 7 – Код підключееня до БД

*self.connection = mysql.connector.connect(host='localhost', database='spammer', user='root', password=pass')*

*if self.connection.is\_connected():*

*cursor = self.connection.cursor()*

*cursor.execute('SELECT email FROM users')*

*self.receivers = cursor.fetchall()*

*cursor.execute('SELECT \* FROM messages')*

*self.messages = cursor.fetchall()*

# 3.3.3 Початок розсилки

Для початку розсилки створимо метод startSending який буде створювати об’єкт класу Spammer та починати розсилку. Код метода показано у Лістинзі 8

Лістинг 8 – Код метода startSending()

def startSending(self):

try:

spammer = Spammer(self.emailEdit.text(), self.passwordEdit.text(), self.receivers)

message = EmailMessage()

message.set\_content(self.messageEdit.toPlainText())

message['Subject'] = self.subjectEdit.text()

print(self.messageEdit.toPlainText())

spammer.run(message)

except:

pass

# 3.4 Запуск програми

Для запуску програми перевіримо чи наш файл називається main і проініціалізуємо усі необхідні об’кти. В нашому випадку це вікно застосунку і запустимо його. Код усього файлу наведено у Лістинзі 9

Лістинг 9 – код файлу main.py

*from PyQt6.QtWidgets import QApplication, QWidget, QGridLayout, QPushButton, QLabel, QLineEdit, QTextEdit, QComboBox*

*import sys*

*from spammer import Spammer*

*from email.message import EmailMessage*

*import mysql.connector*

*class Window(QWidget):*

*def \_\_init\_\_(self):*

*super().\_\_init\_\_()*

*self.setWindowTitle('Spammer')*

*layout = QGridLayout()*

*layout.addWidget(QLabel('Sender email'), 0, 0)*

*self.emailEdit = QLineEdit()*

*self.emailEdit.setText('sendermail@gmail.com')*

*layout.addWidget(self.emailEdit, 0, 1)*

*layout.addWidget(QLabel('Sender password'), 1, 0)*

*self.passwordEdit = QLineEdit()*

*self.passwordEdit.setEchoMode(QLineEdit.EchoMode.Password)*

*self.passwordEdit.setText('senderpass')*

*layout.addWidget(self.passwordEdit, 1, 1)*

*layout.addWidget(QLabel('Subject'), 2, 0)*

*self.subjectEdit = QLineEdit()*

*self.subjectEdit.setText('Promo')*

*layout.addWidget(self.subjectEdit, 2, 1)*

*layout.addWidget(QLabel('Message'), 3, 0)*

*self.messageEdit = QTextEdit()*

*self.messageEdit.setText('Message')*

*layout.addWidget(self.messageEdit, 3, 1)*

*layout.addWidget(QLabel('Choose message'), 4, 0)*

*self.messageChoose = QComboBox()*

*self.messageChoose.currentIndexChanged.connect(self.onMessageChoose)*

*layout.addWidget(self.messageChoose, 4, 1)*

*startSendingBtn = QPushButton('Start')*

*startSendingBtn.clicked.connect(lambda: self.startSending())*

*layout.addWidget(startSendingBtn, 5, 1)*

*self.setLayout(layout)*

*self.connection = mysql.connector.connect(host='localhost', database='spammer', user='root', password='pass')*

*if self.connection.is\_connected():*

*cursor = self.connection.cursor()*

*cursor.execute('SELECT email FROM subscribers')*

*self.receivers = cursor.fetchall()*

*cursor.execute('SELECT \* FROM messages')*

*self.messages = cursor.fetchall()*

*self.subjectEdit.setText(self.messages[0][1])*

*self.messageEdit.setText(self.messages[0][2])*

*print(self.messages)*

*for i in self.messages:*

*self.messageChoose.addItem(i[1])*

*def onMessageChoose(self):*

*self.subjectEdit.setText(self.messages[self.messageChoose.currentIndex()][1])*

*self.messageEdit.setText(self.messages[self.messageChoose.currentIndex()][2])*

*def startSending(self):*

*try:*

*spammer = Spammer(self.emailEdit.text(), self.passwordEdit.text(), self.receivers)*

*message = EmailMessage()*

*message.set\_content(self.messageEdit.toPlainText())*

*message['Subject'] = self.subjectEdit.text()*

*print(self.messageEdit.toPlainText())*

*spammer.run(message)*

*except:*

*pass*

*if \_\_name\_\_ == '\_\_main\_\_':*

*app = QApplication(sys.argv)*

*window = Window()*

*window.show()*

*sys.exit(app.exec())*

# 4 Тестування системи

Для тестування використаймо сайти з тимчасовими скриньками. Далі заповнимо базу даних, як показано на рисунках 4.1 і 4.2



Рисунок 4.2 – Заповнення таблиці Users



Рисунок 4.2 – Заповнення таблиці Messages

Далі запустимо додаток і введемо всі необхідні дані як показано на рисунку 4.3

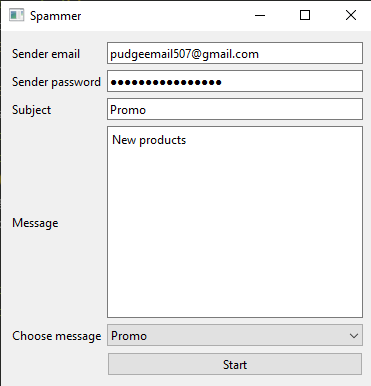


Рисунок 4.3 – Введення даних

Тепер виконаємо розсилку і перевіримо наші скриньки (рисунок 4.4, 4.5)

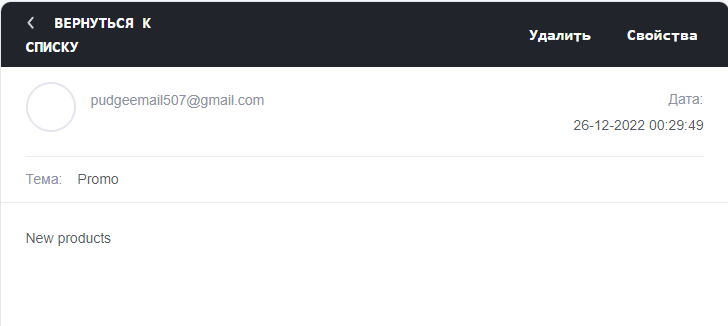


Рисунок 4.4 – Отримання листа на тимчасову пошту

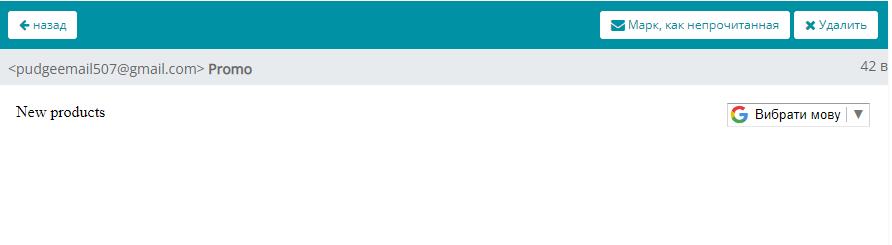


Рисунок 4.5 – Отримання листа на тимчасову пошту

Як бачимо застосунок працює правильно. Більш детальне тестування програми показано на відео, що прикріплено ді звіту.

# Порівняння з аналогами

Порівняємо нашу програму з AMS(Advanced Mass Sender) та LetsExtract Email Studio.

AMS – це вузькоспеціалізована програма для масових розсилок. Має інструменти для створення унікальних повідомлень. Приклад інтерфейса показано на рисунку 4.6

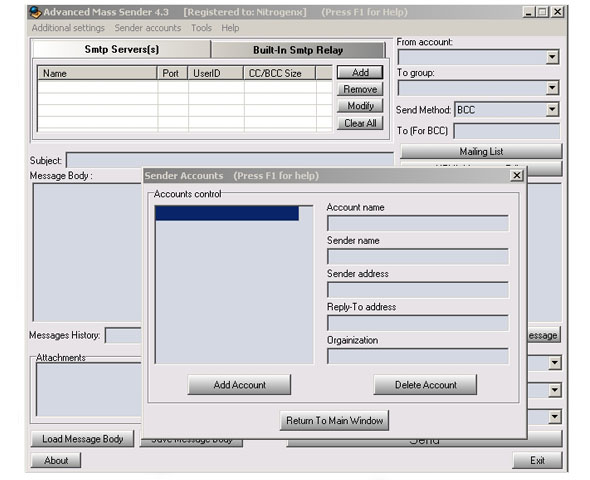


Рисунок 4.6 – Вигляд інтерфейса програми Advanced Mass Sender

Ця програма схожа на створену в курсовому проекті оскільки вона в цілому виконує лише розсилку повідомлень

LetsExtract Email Studio – Потужний інструмент для збирання даних користуваів таких як: email-адреси, номери телефонів та месенджерів. Також дозволяє зібрати базу скринько з відомих сервісі. Також є вбудований валідатор email і звісно інструмент для виконання розсилки. Приклад інтерфейса програми показано на рисунку 4.7

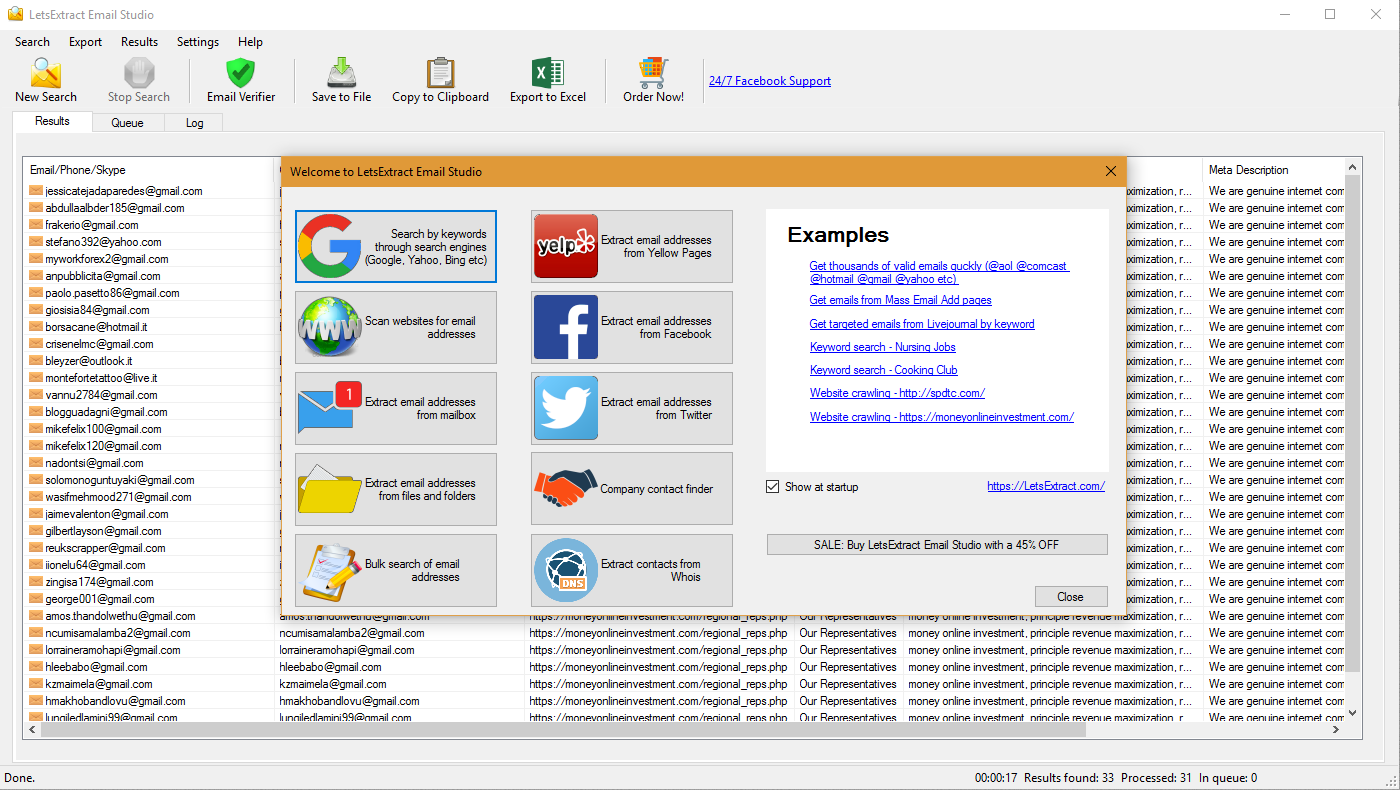


Рисунок 4.7 – Приклад інтерфейса програми LetsExtract Email Studio

Як бачимо даний софт є потужним інструментом не лише для розсилок листів. Його можна було б використати у парі з нашою програмою.

В цілому наша програма виконує основну функцію як і інші, проте вона дуже легка й проста в реалізації а тому їй нема аналогів.

# ВИСНОВКИ

За результатами виконання курсової роботи сформовано такі висновки:

1. Була створена програма, що виконує розсилку рекламних листів на пошту підписників
2. Було створено базу даних з списком підписників і з шаблонами повідомлень. Дана БД симулює справжню БД великого сервісу
3. Під час розробки проекту була досліджена робота бібліотек smtplib та її можливість відправляти листи
4. Завдяки тому, що обраний імітатор створювався на мові Python, він може використовуватися на платформах Linux та MacOS та будь-яких інших платформах, що мають інтерпритатор Python
5. Було вивчено роботу з бібліотекою PyQt для створення графічного компонента
6. У порівнянні з аналогами створена програма задовольняє усі необхідні задачі

# ВИКОРИСТАНІ ДЖЕРЕЛА

1. Офіційна документація PyQt [Електронний ресурс] – Режим доступу до ресурсу: https://doc.qt.io/qtforpython/
2. Офіційна документація з smtplib [Електронний ресурс] – Режим доступу до ресурсу: https://docs.python.org/3/library/smtplib.html.
3. Маттес Е. Пришвидшений курс Python / Ерік Маттес., 2021. – 600 с.
4. Мартін Р. Чиста архітектура / Роберт Мартін., 2019. – 416 с.
5. Васильєв О. Програмування мовою Python / Олексій Васильєв., 2019. – 504 с.
6. Офіційна документація мови Python [Електронний ресурс] – Режим доступу до ресурсу: https://docs.python.org/3/.