|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования  **«Российский технологический университет - МИРЭА»** |
| **РТУ МИРЭА** |

Институт информационных технологий (ИИТ)Кафедра вычислительной техники (ВТ)

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №4**

**по дисциплине**

«Инструментальное программное обеспечение разработки и проектирования информационных систем»

Выполнил студент группы ИКМО-01-20 Баранецкий Д.А.

Фамилия И.О.

Принял Куликов А.А.

Фамилия И.О.

Работы выполнены «\_\_»\_\_\_\_\_\_\_2021 г. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись студента)

«Зачтено» «\_\_»\_\_\_\_\_\_\_2021 г. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись руководителя)

Москва, 2021 г.

Оглавление

[Задание 3](#_Toc66636973)

[Код программы 3](#_Toc66636974)

[Скриншоты 10](#_Toc66636975)

# Задание

**Цель практическое работы:**

Сделать веб сервис для отправки email сообщений.

**Задачи:**

* Создать простое приложение по отправки писем.
* Интерфейс должна передавать следующие параметры на сервер:
  + Почта адресанта
  + Пароль
  + Сервер для пересылки писем
  + Тема письма
  + Текст сообщения для отправки
  + Данные об адресатах для отправки, прочитанные из CSV-файла
* Для разработки веб-сервиса должна использоваться библиотека Flask
* Создать форму в QT Designer

# Код программы

mailInterfaceUI.py – файл, формирующий форму отправки

# -\*- coding: utf-8 -\*-  
  
# Form implementation generated from reading ui file 'mailSenderUI.ui'  
#  
# Created by: PyQt5 UI code generator 5.15.0  
#  
# WARNING: Any manual changes made to this file will be lost when pyuic5 is  
# run again. Do not edit this file unless you know what you are doing.  
  
  
from PyQt5 import QtCore, QtGui, QtWidgets  
  
  
class Ui\_MainWindow(object):  
 def setupUi(self, MainWindow):  
 MainWindow.setObjectName("MainWindow")  
 MainWindow.resize(800, 600)  
  
 self.centralwidget = QtWidgets.QWidget(MainWindow)  
 self.centralwidget.setObjectName("centralwidget")  
  
 self.verticalLayout\_5 = QtWidgets.QVBoxLayout(self.centralwidget)  
 self.verticalLayout\_5.setObjectName("verticalLayout\_5")  
  
 self.horizontalLayout\_2 = QtWidgets.QHBoxLayout()  
 self.horizontalLayout\_2.setObjectName("horizontalLayout\_2")  
  
 self.label\_3 = QtWidgets.QLabel(self.centralwidget)  
 self.label\_3.setObjectName("label\_3")  
 self.horizontalLayout\_2.addWidget(self.label\_3)  
  
 self.ServerComboBox = QtWidgets.QComboBox(self.centralwidget)  
 self.ServerComboBox.setObjectName("ServerComboBox")  
 self.horizontalLayout\_2.addWidget(self.ServerComboBox)  
 self.verticalLayout\_5.addLayout(self.horizontalLayout\_2)  
  
 self.line = QtWidgets.QFrame(self.centralwidget)  
 self.line.setFrameShape(QtWidgets.QFrame.HLine)  
 self.line.setFrameShadow(QtWidgets.QFrame.Sunken)  
 self.line.setObjectName("line")  
 self.verticalLayout\_5.addWidget(self.line)  
  
 self.horizontalLayout = QtWidgets.QHBoxLayout()  
 self.horizontalLayout.setObjectName("horizontalLayout")  
  
 self.verticalLayout\_4 = QtWidgets.QVBoxLayout()  
 self.verticalLayout\_4.setObjectName("verticalLayout\_4")  
  
 self.label\_4 = QtWidgets.QLabel(self.centralwidget)  
 self.label\_4.setObjectName("label\_4")  
 self.verticalLayout\_4.addWidget(self.label\_4)  
  
 spacerItem = QtWidgets.QSpacerItem(10, 10, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)  
 self.verticalLayout\_4.addItem(spacerItem)  
  
 self.label\_5 = QtWidgets.QLabel(self.centralwidget)  
 self.label\_5.setObjectName("label\_5")  
 self.verticalLayout\_4.addWidget(self.label\_5)  
  
 self.horizontalLayout.addLayout(self.verticalLayout\_4)  
 self.verticalLayout\_3 = QtWidgets.QVBoxLayout()  
 self.verticalLayout\_3.setObjectName("verticalLayout\_3")  
  
 self.LoginEnter= QtWidgets.QLineEdit(self.centralwidget)  
 self.LoginEnter.setObjectName("LoginEnter")  
 self.verticalLayout\_3.addWidget(self.LoginEnter)  
  
 # -- создаём поле ввода пароля --  
 self.PasswordEnter = QtWidgets.QLineEdit(self.centralwidget)  
 self.PasswordEnter.setObjectName("PasswordEnter")  
 self.verticalLayout\_3.addWidget(self.PasswordEnter)  
  
 self.horizontalLayout.addLayout(self.verticalLayout\_3)  
 self.verticalLayout\_5.addLayout(self.horizontalLayout)  
 # -------------------------------  
  
 self.line = QtWidgets.QFrame(self.centralwidget)  
 self.line.setFrameShape(QtWidgets.QFrame.HLine)  
 self.line.setFrameShadow(QtWidgets.QFrame.Sunken)  
 self.line.setObjectName("line")  
 self.verticalLayout\_5.addWidget(self.line)  
  
 self.verticalLayout\_2 = QtWidgets.QVBoxLayout()  
 self.verticalLayout\_2.setObjectName("verticalLayout\_2")  
  
 #-- создаём блок с добавлением файла --  
 self.horizontalLayout\_6 = QtWidgets.QHBoxLayout()  
 self.horizontalLayout\_6.setObjectName("horizontalLayout\_6")  
  
 self.label\_6 = QtWidgets.QLabel(self.centralwidget)  
 self.label\_6.setObjectName("label\_6")  
 self.horizontalLayout\_6.addWidget(self.label\_6)  
  
 self.OpenCSVFile = QtWidgets.QPushButton(self.centralwidget)  
 self.OpenCSVFile.setObjectName("OpenCSVFile")  
 self.horizontalLayout\_6.addWidget(self.OpenCSVFile)  
 self.verticalLayout\_2.addLayout(self.horizontalLayout\_6)  
 self.verticalLayout\_5.addLayout(self.verticalLayout\_2)  
 # --------------------------------------  
  
 self.line = QtWidgets.QFrame(self.centralwidget)  
 self.line.setFrameShape(QtWidgets.QFrame.HLine)  
 self.line.setFrameShadow(QtWidgets.QFrame.Sunken)  
 self.line.setObjectName("line")  
 self.verticalLayout\_5.addWidget(self.line)  
  
 # -- создаём блок с добавлением темы сообщения --  
 self.horizontalLayout\_3 = QtWidgets.QHBoxLayout()  
 self.horizontalLayout\_3.setObjectName("horizontalLayout\_3")  
  
 self.label\_2 = QtWidgets.QLabel(self.centralwidget)  
 self.label\_2.setObjectName("label\_2")  
 self.horizontalLayout\_3.addWidget(self.label\_2)  
  
 self.ThemeBar = QtWidgets.QLineEdit(self.centralwidget)  
 self.ThemeBar.setObjectName("ThemeBar")  
 self.horizontalLayout\_3.addWidget(self.ThemeBar)  
 self.verticalLayout\_5.addLayout(self.horizontalLayout\_3)  
 # -----------------------------------------------  
  
 self.verticalLayout = QtWidgets.QVBoxLayout()  
 self.verticalLayout.setObjectName("verticalLayout")  
  
 # -- создаём поле для ввода сообщения --  
 self.label = QtWidgets.QLabel(self.centralwidget)  
 self.label.setObjectName("label")  
 self.verticalLayout.addWidget(self.label)  
  
 self.textEdit = QtWidgets.QTextEdit(self.centralwidget)  
 self.textEdit.setObjectName("textEdit")  
 self.verticalLayout.addWidget(self.textEdit)  
 # --------------------------------------  
  
 # -- создаём кнопку отправки --  
 self.SendButton = QtWidgets.QPushButton(self.centralwidget)  
 self.SendButton.setObjectName("SendButton")  
 self.verticalLayout.addWidget(self.SendButton)  
 self.verticalLayout\_5.addLayout(self.verticalLayout)  
 # -----------------------------  
  
 MainWindow.setCentralWidget(self.centralwidget)  
  
 self.menubar = QtWidgets.QMenuBar(MainWindow)  
 self.menubar.setGeometry(QtCore.QRect(0, 0, 330, 18))  
 self.menubar.setObjectName("menubar")  
 MainWindow.setMenuBar(self.menubar)  
  
 self.statusbar = QtWidgets.QStatusBar(MainWindow)  
 self.statusbar.setObjectName("statusbar")  
 MainWindow.setStatusBar(self.statusbar)  
  
 self.retranslateUi(MainWindow)  
 QtCore.QMetaObject.connectSlotsByName(MainWindow)  
  
 def retranslateUi(self, MainWindow):  
 \_translate = QtCore.QCoreApplication.translate  
 MainWindow.setWindowTitle(\_translate("MainWindow", "MainWindow"))  
  
 self.OpenCSVFile.setText(\_translate("MainWindow", "Выбрать файл"))  
 self.SendButton.setText(\_translate("MainWindow", "Отправить"))  
 self.label.setText(\_translate("MainWindow", "Сообщение "))  
 self.label\_2.setText(\_translate("MainWindow", "Тема"))  
 self.label\_3.setText(\_translate("MainWindow", "Сервер"))  
 self.label\_4.setText(\_translate("MainWindow", "Адрес Отправителя"))  
 self.label\_5.setText(\_translate("MainWindow", "Пароль"))  
 self.label\_6.setText(\_translate("MainWindow", "Выбрать CSV файл"))  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 import sys  
 app = QtWidgets.QApplication(sys.argv)  
 MainWindow = QtWidgets.QMainWindow()  
 ui = Ui\_MainWindow()  
 ui.setupUi(MainWindow)  
 MainWindow.show()  
 sys.exit(app.exec\_())

main.py – файл, отвечающий за отправку сообщений из формы

import smtplib  
from email.mime.multipart import MIMEMultipart  
from email.mime.text import MIMEText  
from email.mime.image import MIMEImage  
from email.mime.audio import MIMEAudio  
from email.mime.base import MIMEBase  
import email.encoders as encoders  
from PyQt5 import QtWidgets  
import mailSenderUI  
import sys  
import os  
import mimetypes  
import csv  
import requests  
import json  
  
  
class MailAPP(QtWidgets.QMainWindow, QtWidgets.QFileDialog, mailSenderUI.Ui\_MainWindow):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.setupUi(self)  
 self.setWindowTitle('Почта')  
 self.setFocus()  
 self.myEMAIL = ''  
 self.myPassword = ''  
 self.PasswordEnter.setEchoMode(QtWidgets.QLineEdit.Password)  
 self.ThemeBar.setPlaceholderText('Без темы')  
 services = ['gmail.com', 'mail.ru', 'yandex.ru']  
 self.servers = {'gmail.com': 'smtp.gmail.com:587', 'mail.ru': 'smtp.yandex.ru:467',  
 'yandex.ru': 'smtp.mail.ru:25'}  
 for service in services:  
 self.ServerComboBox.addItem(service)  
 self.SendButton.clicked.connect(self.sendMail) # self.sendMail  
 self.OpenCSVFile.clicked.connect(self.selectCSV)  
 self.csvFile = ''  
  
 # отправка сообщения  
 def sendMail(self):  
 if self.LoginEnter.text().split('@')[1] != self.ServerComboBox.currentText():  
 self.showMsgBox('Введенная почта и сервер не совпадают!')  
 else:  
 self.myEMAIL = self.LoginEnter.text()  
 self.myPassword = self.PasswordEnter.text()  
 subject = self.ThemeBar.text()  
 url = 'http://127.0.0.1:1516'  
 data = {'mail': self.myEMAIL,  
 'password': self.myPassword,  
 'subject': subject,  
 'message': self.textEdit.toPlainText(),  
 'csv': self.csvReader(),  
 'server': self.servers[self.ServerComboBox.currentText()]}  
 print(json.dumps(data))  
 requests.post(url, data=json.dumps(data))  
  
  
 # выбор CSV-файла для работы с ним  
 def selectCSV(self):  
 fileName = QtWidgets.QFileDialog.getOpenFileName(parent=self,  
 caption=u'Открыть файл',  
 directory='E:/',  
 filter='csv (\*.csv);;AllFiles (\*)')[0]  
 if fileName: # Если выбран файл  
 self.csvFile = fileName  
 self.label\_6.setText(os.path.basename(self.csvFile))  
  
 # чтение CSV-файла для работы с ним  
 def csvReader(self):  
 receiver = {}  
 with open(self.csvFile, 'r') as csvFile:  
 csvReader = csv.reader(csvFile)  
 for row in csvReader:  
 to = row[0].split(';')[0]  
 file = row[0].split(';')[1:]  
 receiver[to] = file  
 print(receiver)  
 return receiver  
  
 # работа с типами файлов внутри CSV  
 def processAttachment(self, files: list) -> list:  
 filesList = []  
 for file in files:  
 fileName = ''  
 try:  
 fileName = os.path.basename(file)  
 ctype, encoding = mimetypes.guess\_type(file)  
 if ctype is None or encoding is not None:  
 ctype = 'application/octet-stream'  
 maintype, subtype = ctype.split('/', 1)  
 if maintype == 'text':  
 with open(file) as fp:  
 file = MIMEText(fp.read(), \_subtype=subtype)  
 fp.close()  
 elif maintype == 'image':  
 with open(file, 'rb') as fp:  
 file = MIMEImage(fp.read(), \_subtype=subtype)  
 fp.close()  
 elif maintype == 'audio':  
 with open(file, 'rb') as fp:  
 file = MIMEAudio(fp.read(), \_subtype=subtype)  
 fp.close()  
 else:  
 with open(file, 'rb') as fp:  
 file = MIMEBase(maintype, subtype)  
 file.set\_payload(fp.read())  
 fp.close()  
 encoders.encode\_base64(file)  
 file.add\_header('Content-Disposition', 'attachment', filename=fileName)  
 filesList.append(file)  
 except:  
 self.showMsgBox('Ошибка добавления файла: ' + fileName)  
 return filesList  
  
 # вывод окна с сообщением  
 def showMsgBox(self, msg: str):  
 msgBox = QtWidgets.QMessageBox()  
 msgBox.setText(msg)  
 msgBox.exec\_()  
  
  
  
def main():  
 app = QtWidgets.QApplication(sys.argv)  
 window = MailAPP()  
 window.show()  
 sys.exit(app.exec\_())  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()

Web-Service/main.py – файл, отвечающий за отправку сообщений из формы

from flask import Flask, request, jsonify  
import smtplib  
from email.mime.multipart import MIMEMultipart  
from email.mime.text import MIMEText  
from email.mime.image import MIMEImage  
from email.mime.audio import MIMEAudio  
from email.mime.base import MIMEBase  
import email.encoders as encoders  
import os  
import mimetypes  
  
app = Flask(\_\_name\_\_)  
host = '127.0.0.1'  
port = 1516  
  
  
def processAttachment(files: list) -> list:  
 filesList = []  
 for file in files:  
 fileName = ''  
 try:  
 fileName = os.path.basename(file)  
 ctype, encoding = mimetypes.guess\_type(file)  
 if ctype is None or encoding is not None:  
 ctype = 'application/octet-stream'  
 maintype, subtype = ctype.split('/', 1)  
 if maintype == 'text':  
 with open(file) as fp:  
 file = MIMEText(fp.read(), \_subtype=subtype)  
 fp.close()  
 elif maintype == 'image':  
 with open(file, 'rb') as fp:  
 file = MIMEImage(fp.read(), \_subtype=subtype)  
 fp.close()  
 elif maintype == 'audio':  
 with open(file, 'rb') as fp:  
 file = MIMEAudio(fp.read(), \_subtype=subtype)  
 fp.close()  
 else:  
 with open(file, 'rb') as fp:  
 file = MIMEBase(maintype, subtype)  
 file.set\_payload(fp.read())  
 fp.close()  
 encoders.encode\_base64(file)  
 file.add\_header('Content-Disposition', 'attachment', filename=fileName)  
 filesList.append(file)  
 except:  
 print('Ошибка добавления файла: ' + fileName)  
 return filesList  
  
  
# @app.route('/sender/<str:server>/<str:mail>/<str:password>/<dict:csvData>/<str:text>', methods=['GET', 'POST'])  
@app.route('/', methods=['GET', 'POST'])  
# def index(mail: str, password: str, csvData: dict, text: str):  
def index():  
 if request.method == 'POST':  
 content = request.get\_json(force=True)  
 mail = content['mail']  
 password = content['password']  
 subject = content['subject']  
 message = content['message']  
 server = content['server']  
 csv = content['csv']  
 print(mail, server, password, subject, message, csv)  
 smtpServer = smtplib.SMTP(server) # Создание объекта SMTP  
 smtpServer.starttls() # Шифрование  
 try:  
 smtpServer.login(mail, password) # Ввод учетных данных  
 except:  
 print('Ошибка ввода учетных данных')  
 else:  
 for k, v in csv.items():  
 print('k', k)  
 print('v', v)  
 myMessage = MIMEMultipart() # Создание объекта стандарта MIME  
 myMessage['Subject'] = subject # Тема сообщения  
 myMessage['From'] = mail # Отправитель  
 myMessage['To'] = k # Получатель  
 messageContent = message # Получение текста с формы  
 myMessage.attach(MIMEText(messageContent, 'plain')) # Добавление текста в сообещение  
 for file in processAttachment(v):  
 print('file', v)  
 myMessage.attach(file)  
 try:  
 smtpServer.send\_message(myMessage) # Отправка сообщения  
 except:  
 print(f'Не удалость отправить сообщение адресату {k}')  
 return ''  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 app.run(host=host, port=port, debug=True)

# Скриншоты

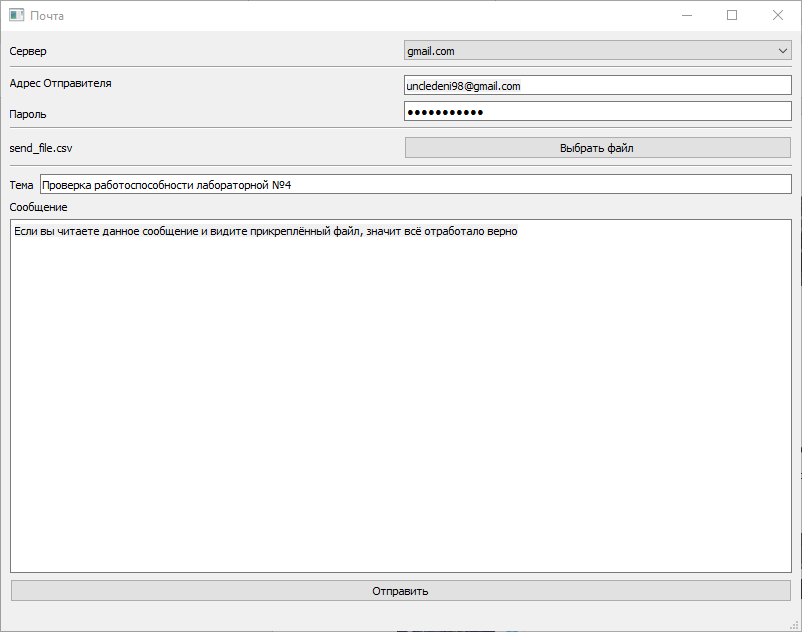


Рис.1. – Интерфейс и отправка сообщения на почты с CSV-файле



Рис.2. – Содержимое CSV-файла

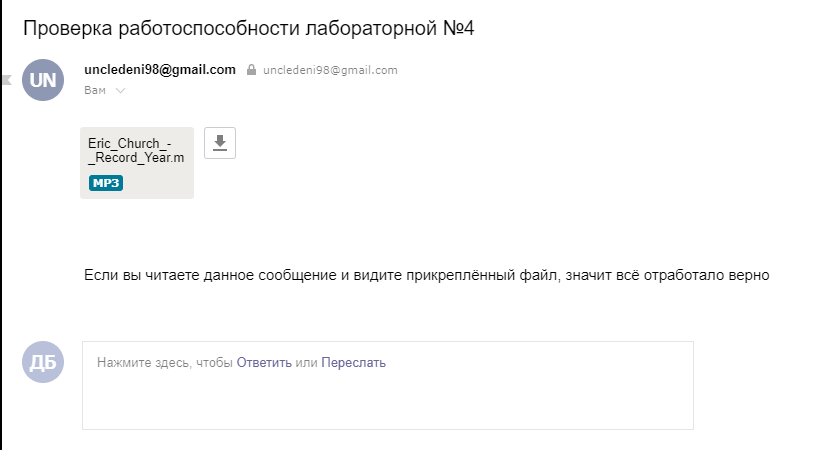


Рис.3. – Полученное письмо. Яндекс почта

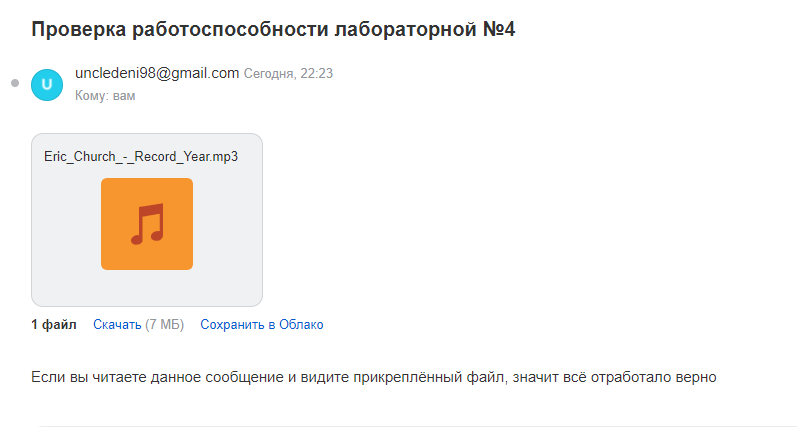


Рис.4. – Полученное письмо. Mail.ru