МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ государственное БЮДЖЕТНОЕ

образовательное учреждение

высшего образования

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кафедра защиты информации

**

**ОТЧЁТ**

**по практической работе №4**

**«**Разработка сервиса статистики**»**

**по дисциплине: «***Программирование***»**

Выполнил:Проверил:

Студент гр. «АБс-222», «АВТФ» *доцент кафедры ЗИ*

*Азява Д.А. Архипова А. Б.*

«\_\_\_» \_\_\_\_\_\_ 2023 г.«\_\_\_» \_\_\_\_\_\_ 2023 г.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись) (подпись)

Новосибирск 2023

Цели и задачи работы: Требуется разработать сервис агрегации статистики переходов по сокращенным ссылкам.

Сервис статистики должен собирать статистику с сервиса сокращения ссылок о переходах по сокращённым ссылкам и генерировать отчеты по запросу пользователя.

Таблица 1 – Код программы

|  |  |
| --- | --- |
| manger.py | import json  def generate\_html\_report(filename, column\_order, html\_file):  # Чтение данных из JSON файла  with open(filename, 'r') as file:  data = json.load(file)  # Создание словарей для хранения количества повторений URL, IP и Time  url\_count = {}  ip\_count = {}  time\_count = {}    # Подсчет количества повторений URL, IP и Time  for entry in data:  url = entry["URL"]  ip = entry["IP"]  time = entry["Time"]    url\_count[url] = url\_count.get(url, 0) + 1  ip\_count[ip] = ip\_count.get(ip, 0) + 1  time\_count[time] = time\_count.get(time, 0) + 1    # Создание HTML таблицы  html = "<h1 align='center'>Отчёт</h1><table align='center' border='1'>"  html += "<tr>"    # Создание заголовков таблицы в заданном пользователем порядке  for column in column\_order:  html += f"<th>{column}</th>"    html += "</tr>"    # Вывод уникальных значений URL, IP и Time и их количества повторений  for url, count in url\_count.items():  html += "<tr>"    # Вывод URL, IP и Time только один раз для каждого уникального значения  html += f"<td rowspan='{count}'>{url}</td>"    ips = [entry["IP"] for entry in data if entry["URL"] == url]  unique\_ips = list(set(ips))    for i, ip in enumerate(unique\_ips):  if i > 0:  html += "<tr>"    # Вывод IP и соответствующего времени для каждого уникального значения  html += f"<td rowspan='{ips.count(ip)}'>{ip}</td>"    times = [entry["Time"] for entry in data if entry["URL"] == url and entry["IP"] == ip]  unique\_times = list(set(times))    for j, time in enumerate(unique\_times):  if j > 0:  html += "<tr>"    html += f"<td>{time}</td>"  html += "</tr>"  html += f'<tr><td>{count}</td><td>{ips.count(ip)}</td><td>{enumerate(unique\_times)}</td></tr>'    html += "</table>"  # Сохраняем HTML отчет в файл  with open(html\_file, 'w') as f:  f.write(html)  # Пример использования  # Порядок столбцов  column\_order = ["URL", "IP", "Time"]  generate\_html\_report("statistic.json", column\_order, "test.html") |
| service.py | from flask import Flask, render\_template, request, redirect  from urllib.parse import quote  import socket  import datetime  import json  class JSONCreator:  def \_\_init\_\_(self, filename):  self.filename = filename  self.data = self.load\_data()  def load\_data(self):  try:  with open(self.filename, 'r') as file:  return json.load(file)  except (FileNotFoundError, json.JSONDecodeError):  return []  def add\_data(self, url, ip, time):  if isinstance(time, datetime.datetime):  time = time.strftime('%Y-%m-%d %H:%M:%S')  data\_entry = {"URL": url, "IP": ip, "Time": time}  self.data.append(data\_entry)  def create\_json(self):  with open(self.filename, 'w') as file:  json.dump(self.data, file)  def reduction(text):  # Добиваем строку, до кратности 8-ми  while ((len(text) % 8) != 0):  text += "#"    output = ['\0', '\0', '\0', '\0', '\0', '\0', '\0', '\0']  # Определяем размер блоков  length = len(text) // 8  for i in range(0, 9):  res = 0  for j in range(0, (length \* i)):  res += ord(text[j]) \* ord(text[j]) \* j  res %= 1000  resulto = 0  while (res > 0):  resulto += res % 10  res //= 10    resulto += 65  if resulto > 90:  resulto -= 4  output[i - 1] = chr(resulto)  return ''.join(output)  app = Flask(\_\_name\_\_)  server\_address = ('127.0.0.1', 5656)  @app.route('/', methods=['GET', 'POST'])  def home():  generated\_link = None  if request.method == 'POST':  original\_link = request.form['user\_input']  short\_link = reduction(original\_link)  # Сохраняем короткую ссылку в базе данных  with socket.socket(socket.AF\_INET, socket.SOCK\_STREAM) as s:  try:  s.connect(server\_address)  s.sendall(f"PUSH#{short\_link}#{original\_link}\0'".encode())  print("Message sent successfully.")  data = s.recv(1024)  print(f"Response from server: {data.decode()}")  s.sendall(f"end".encode())  except ConnectionRefusedError:  print("Connection to the server failed.")  # Возвращаем короткую ссылку  generated\_link = f"http://127.0.0.1:5000/{short\_link}"  return render\_template('index.html', output\_link=generated\_link)  @app.route('/<short\_link>')  def redirect\_to\_original(short\_link):  if short\_link == 'favicon.ico':  return "Not Found"    # Получаем оригинальную ссылку из базы данных  with socket.socket(socket.AF\_INET, socket.SOCK\_STREAM) as s:  try:  s.connect(server\_address)  s.sendall(f"GET#{short\_link}\0".encode())  data = s.recv(1024)  original\_link = data.decode()  #print("catch: " + original\_link + "|end")  s.sendall(f"end".encode())  except ConnectionRefusedError:  print("Connection to the server failed.")  return "Internal Server Error"  # Открытие оригинальной ссылки  if original\_link == "NOT\_FOUND":  return render\_template('!index.html')  else:  json\_creator = JSONCreator('statistic.json')  json\_creator.add\_data(original\_link + "(" + short\_link+ ")", request.environ['REMOTE\_ADDR'], datetime.datetime.now().replace(microsecond=0))  json\_creator.create\_json()  return redirect(original\_link)  if \_\_name\_\_ == '\_\_main\_\_':  app.run(host='127.0.0.1', port=5000, debug=True) |

В результате работы сервиса на Python в случае, когда была от сервера была получена правильная ссылка, формируется JSON объект и записывается и дозаписывается в JSON-файл (Рисунок 1), затем осуществляется переход на запрашиваемый интернет ресурс.

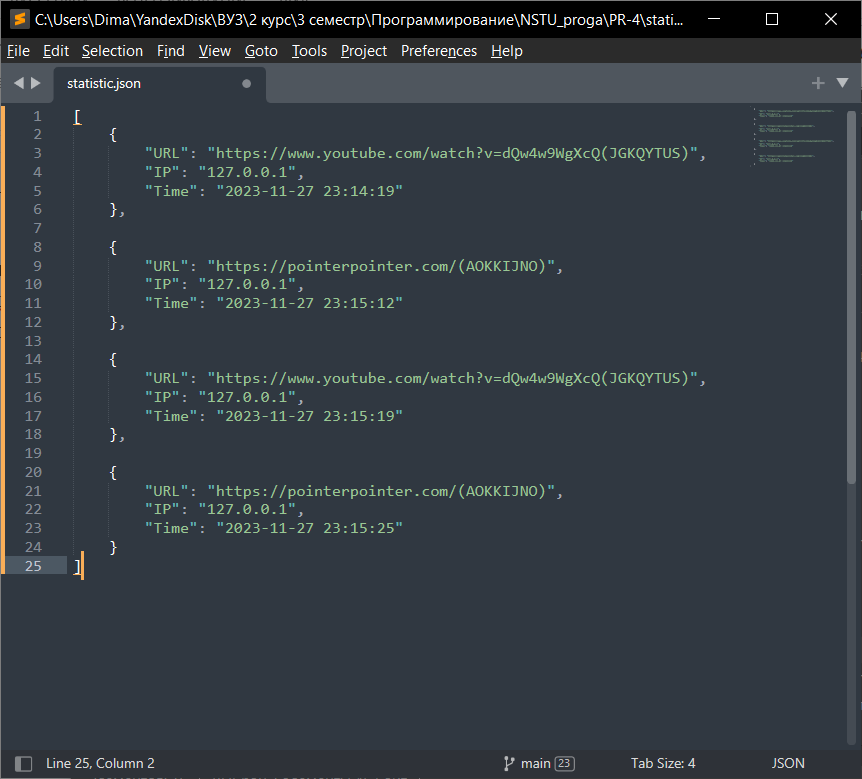


Рисунок 1 – Содержимое JSON-файла

Для формирования HTML-отчёта необходимо запустить manager.py, в результате его работы будет сформирован файл test.html (Рисунок 2).

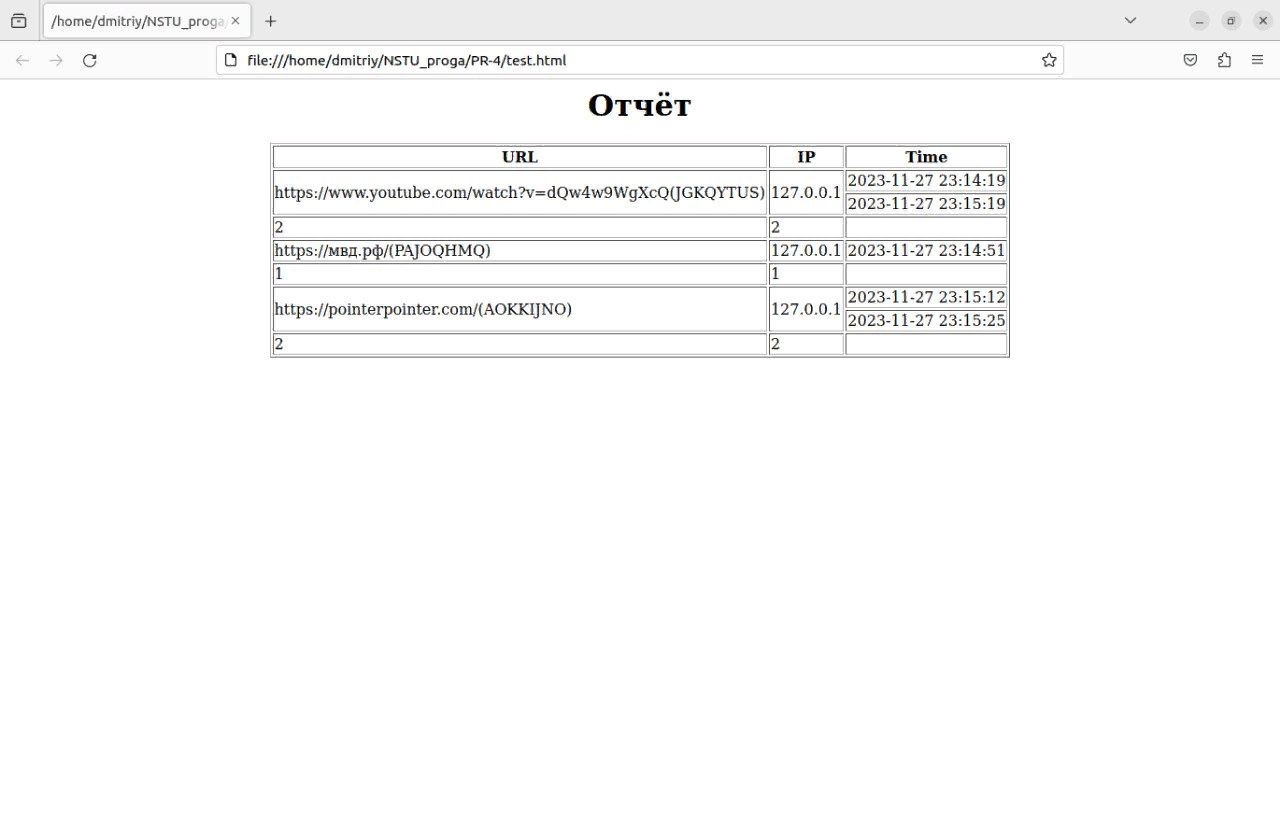


Рисунок 2 – Сформированный HTML-отчёт

Вывод: в результате выполнения лабораторной работы были получены навыки по формированию JSON-файл и формированию HTML-отчёт на основе этих JSON-файлов.