МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФГБОУ ВО

«Пермский государственный национальный исследовательский университет»

Механико-математический факультет

Кафедра информационной безопасности и систем связи

**Отчёт**

по лабораторной работе №1 «Создание клиентского приложения по выполнению CRUD операций на популярном информационном ресурсе»

по дисциплине «Технологии разработки распределенных приложений»

Работу выполнил Проверил

студент гр. КМБ-16 доцент кафедры

Якин Никита прикладной математики

Николаевич и информатики

29.09.2020 Деменев Алексей

Геннадьевич

Пермь, 2020 г

**Содержание**

[**Постановка задачи** 3](#_Toc53921804)

[**Ход выполнения работы** 4](#_Toc53921805)

[**Описание приложения** 4](#_Toc53921806)

[**Заключение** 7](#_Toc53921807)

[**Используемые в ходе работы ресурсы** 8](#_Toc53921808)

[**Код программы** 9](#_Toc53921809)

# **Постановка задачи**

***Цель:*** Изучение возможностей использования API современных популярных информационных ресурсов внешними приложениями.

***Формируемая компетенция:*** способность применять на практике теоретические основы и общие принципы разработки распределенных систем.

***Организация выполнения работы:*** Каждый студент выполняет индивидуальное задание.

***Требования к выполнению работы:***

1. Разработанное приложение должно предоставлять возможности создания, чтения, обновления и удаления некоторых объектов выбранного информационного ресурса (выполнения CRUD‑операций).
2. CRUD‑операции должны выполняться при помощи API выбранного ресурса, причём по крайней мере один тип операции должен требовать аутентификации на нём.
3. Должна быть возможность выполнения по крайней мере двух CRUD‑операций для одного и того же объекта выбранного информационного ресурса.

| **Требование к заданию** | **Максимальное количество баллов** |
| --- | --- |
| Приложение позволяет выполнять CRUD‑операции над объектами выбранного ресурса | 1 (за каждый тип операции) |
| Приложение позволяет выполнять по крайней мере две CRUD‑операции для одного и того же объекта выбранного ресурса | 1 |
| Приложение выполняет по крайней мере одну из CRUD‑операций после успешной аутентификации | 2 |
| Приложение не требует повторной аутентификации при перезапуске программы | 1 |
| Параметры успешной аутентификации сохраняются при перезапуске программы в зашифрованном виде | 2 |

# **Ход выполнения работы**

В качестве ресурса была выбрана социальная сеть Вконтакте. В качестве объекта, над которым будут выполняться CRUD-операции была выбрана «Стена» – это специальное пространство, где человек может размещать любую информацию.

Код программы написан на python[1]. В качестве IDE использовался PyCharm Community Edition [2]. PyCharm Community Edition является бесплатной версией, обладающей усеченным набором возможностей.

Для работы с API Вконтакте [3] использовалась open-source библиотека VK [4].

Для шифрования данных для авторизации использовалась библиотека PyCryptodome [5].

# **Описание приложения**

Приложение позволяет авторизированному пользователю выполнять следующие операции:

* Create - Позволяет создать запись на стене;
* Read - Возвращает список записей со стены пользователя;
* Update - Редактирует запись на стене;
* Delete - Удаляет запись со стены.

При запуске приложение просматривает файл authData.txt, если такого файла нет, то приложение запрашивает у пользователя данные для аутентификации. При успешной авторизации данные сохраняются в файле authData. authData содержит информацию лишь об одном аккаунте. Данные для авторизации конкатенируются и шифруются алгоритмом симметричного шифрования DES.

Содержимое файла authData.txt:

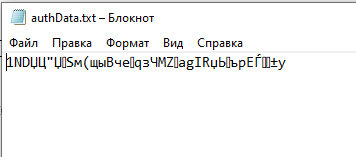


Рис. 1 Файл AuthData.txt

Если файл authData отсутствует, приложение предлагает авторизоваться.

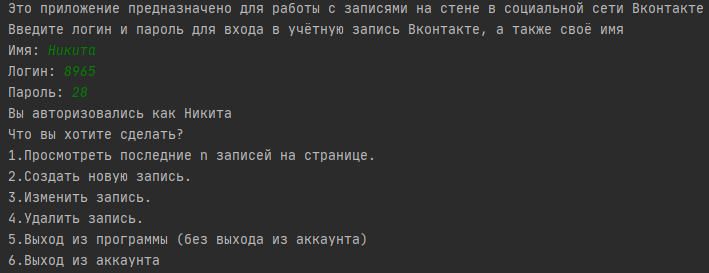


Рис. 2 Процесс авторизации

При успешной авторизации пользователю представляются следующие варианты дальнейшей работы:

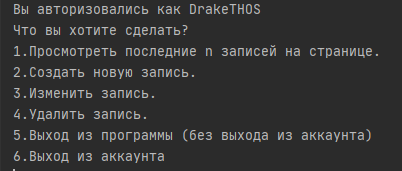


Рис. 3 Список функций

* 1. Read. Просмотр последних записей.

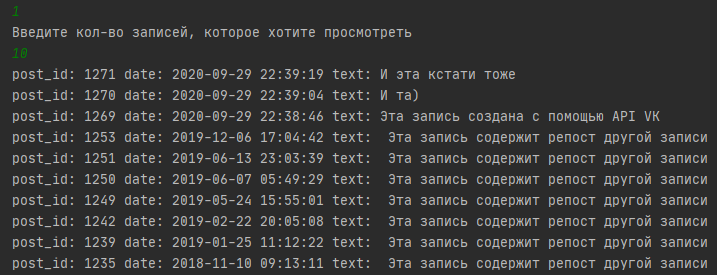


Рис. 4 Результат функции Read

* 1. Create. Создание новой записи.

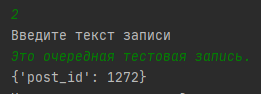


Рис. 5 Результат функции Create

* 1. Edit. Изменение записи.

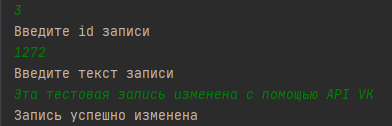


Рис. 6 Результат функции Edit

Все записи можно просмотреть на «Стене» в социальной сети Вконтакте.

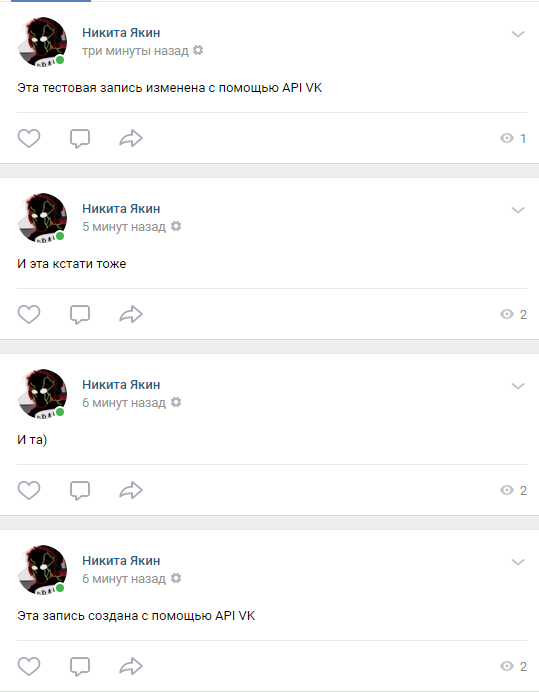


Рис. 7 Результат выполнения операций

* 1. Delete. Удаление записи.

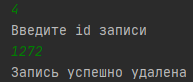


Рис. 8 Результат функции Delete

Запись удалена.

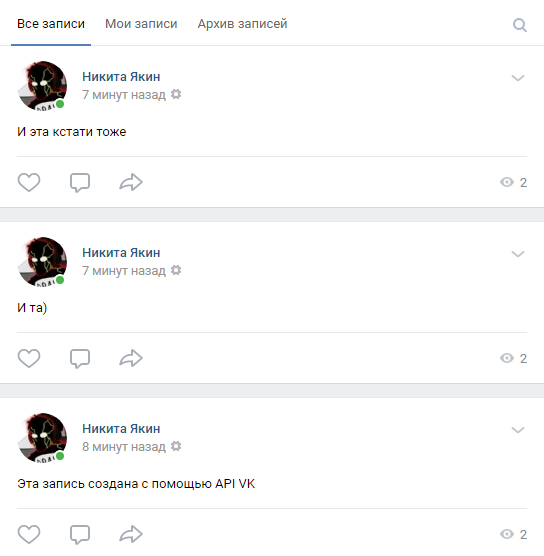


Рис. 9 Результат функции Delete 2

* 1. Выход из программы (без выхода из аккаунта).

При выборе данной опции пользователь выходит из приложения, а его данные сохраняются в authData.txt, при следующем запуске приложения он автоматически авторизуется.

* 1. Выход из аккаунта.

При выборе данной опции пользователь выходит из приложения, а файл с его данными удаляется.

# **Заключение**

В ходе работы достигнута цель и выполнены следующие требования:

Приложение позволяет выполнять CRUD операции над объектами выбранного ресурса. Приложение позволяет выполнять 4 операции (create, read, update, delete) над записями на стене Вконтакте. (4 балла)

Приложение позволяет выполнять по крайней мере две CRUD операции для одного и того же объекта выбранного ресурса. К одной и той же записи в примерах применялись все 4 операции. (1 балл)

Приложение выполняет по крайней мере одну из CRUD операций после успешной аутентификации. Операции можно выполнять лишь после авторизации. (2 балла)

Приложение не требует повторной аутентификации при перезапуске программы. Данные для аутентификации сохраняются в отдельном файле. (1 балл)

Параметры успешной аутентификации сохраняются при перезапуске программы в зашифрованном виде. Данные для аутентификации шифруются DES и сохраняются в отдельном файле. (2 балла)

Итого: 10 баллов

# **Используемые в ходе работы ресурсы**

* + 1. Лицензия Python, <https://docs.python.org/3.7/license.html>
    2. PyCharm Community Edition. Распространяется под лицензией Apache 2, URL: <https://www.jetbrains.com/pycharm/features/editions_comparison_matrix.html> (Дата обращения 29.09.2020)
    3. Документация VK API, URL: <https://vk.com/dev/manuals> (Дата обращения 29.09.2020)
    4. VK. Open-source библиотека, разработанная Дмитрием Ворониным, URL: <https://github.com/voronind/vk> (Дата обращения 29.09.2020)
    5. Библиотека PyCryptodome, URL: <https://pycryptodome.readthedocs.io/en/latest/src/license.html> (Дата обращения 29.09.2020)

# **Код программы**

|  |
| --- |
| import os import vk from Cryptodome.Cipher import DES import datetime  key = b'abcdefgh' des = DES.new(key, DES.MODE\_ECB) filePath = "D:\\authData.txt" value = 1   def pad(text):  while len(text) % 8 != 0:  text += b' '  return text   def createPost():  message = input("Введите текст записи\n")  response = vkApi.wall.post(message=message)  print(response)  return response   def deletePost():  postId = input("Введите id записи\n")  response = vkApi.wall.delete(post\_id=postId)  if response == 1:  print('Запись успешно удалена')  else:  print('Произошла ошибка при удалении сообщения')  return response   def getLastPosts():  posts = input("Введите кол-во записей, которое хотите просмотреть\n")  response = vkApi.wall.get(count=posts)  items = response['items']  for item in items:  date = datetime.datetime.fromtimestamp(item['date']).strftime('%Y-%m-%d %H:%M:%S')  try:  if item['copy\_history']:  text = 'Эта запись содержит репост другой записи'  else:  text = ''  except Exception:  text = ''  print('post\_id: '+str(item['id'])+' date: '+date+' text: '+item['text']+' ' + text)  return response   def editPost():  postId = input("Введите id записи\n")  message = input("Введите текст записи\n")  response = vkApi.wall.edit(post\_id=postId, message=message)  if response == 1:  print('Запись успешно изменена')  else:  print('Произошла ошибка при изменении сообщения сообщения')  return response   def exitProgram():  print('Завершение работы')   def exitAcc():  os.remove(filePath)  print('Завершение работы')   print('Это приложение предназначено для работы с записями на стене в социальной сети Вконтакте') if(os.path.isfile(filePath)):  with open(filePath, 'rb') as f:  encryptedData = f.read()  decryptedData = des.decrypt(encryptedData)  data = decryptedData.decode('utf-8').split('!')  name = data[0]  login = data[1]  password = data[2].replace(' ', '') else:  print('Введите логин и пароль для входа в учётную запись Вконтакте, а также своё имя')  name = input("Имя: ")  login = input("Логин: ")  password = input("Пароль: ")  data = name+'!'+login+'!'+password  encryptedData = des.encrypt(pad(bytes(data, 'utf-8')))  with open(filePath, 'wb') as f:  f.write(encryptedData) try:  session = vk.AuthSession(appId, login, password, scope='wall')  vkApi = vk.API(session, v='5.35', lang='ru')  print("Вы авторизовались как " + name) except Exception:  print('Произошла ошибка при попытке авторизации, проверьте введённые данные')  os.remove(filePath)  value = -1  while value > 0 and value < 5:  value = int(input("Что вы хотите сделать?\n1.Просмотреть последние n записей на странице.\n2.Создать новую запись.\n3.Изменить запись.\n4.Удалить запись.\n5.Выход из программы (без выхода из аккаунта)\n6.Выход из аккаунта\n"))  result = {  1: getLastPosts,  2: createPost,  3: editPost,  4: deletePost,  5: exitProgram,  6: exitAcc,  }.get(value, exitProgram)() |