**TALLINNA TÖÖSTUSHARIDUSKESKUS**

**«Свой проект в Python. Система регистрации и авторизации и её дальнейшее использование»  
Зачетная работа по основам программирования**

**Ученик: Maria Buštšik  
Учитель: Marina Oleinik**

**Tallinn  
2021**

# Введение

Актуальностью выбора темы проекта «Система регистрации и авторизации» является то, что данная система является одной из самых популярных и необходимых частей любой онлайн системы или приложения.

Регистрация и авторизация - за этими сложными терминами скрываются действия, выполняемые ежедневно большинством пользователей машинально.

При входе в онлайн систему или электронную почту система безопасности сайта просит ввести учетные данные: логин и пароль. В обмен на верную пару логин-пароль мы получаем доступ к своему личному кабинету или электронной почте.

Надёжный пароль является одним из самых главных компонентов безопасности хранения данных. В нашем проекте мы применим функцию генерации случайного пароля, подобрать который стороннему пользователю практически невозможно.

Для создания данного проекта мы будем использовать программу Visual Studio и импортируемый модуль графического интерфейса языка Python TKinter.

Tkinter (от англ. tk interface) - это графическая библиотека, позволяющая создавать программы с оконным интерфейсом. Эта библиотека является интерфейсом к популярному языку программирования и инструменту создания графических приложений.

Все виджеты в Tkinter обладают некоторыми общими свойствами. Виджеты создаются вызовом конструктора соответствующего класса.

Последовательность шагов при создании графического приложения имеет свои особенности. Программа должна выполнять свое основное назначение, быть удобной для пользователя, реагировать на его действия. Мы не будем вдаваться в подробности разработки, а рассмотрим какие этапы приблизительно нужно пройти при программировании, чтобы получить программe:

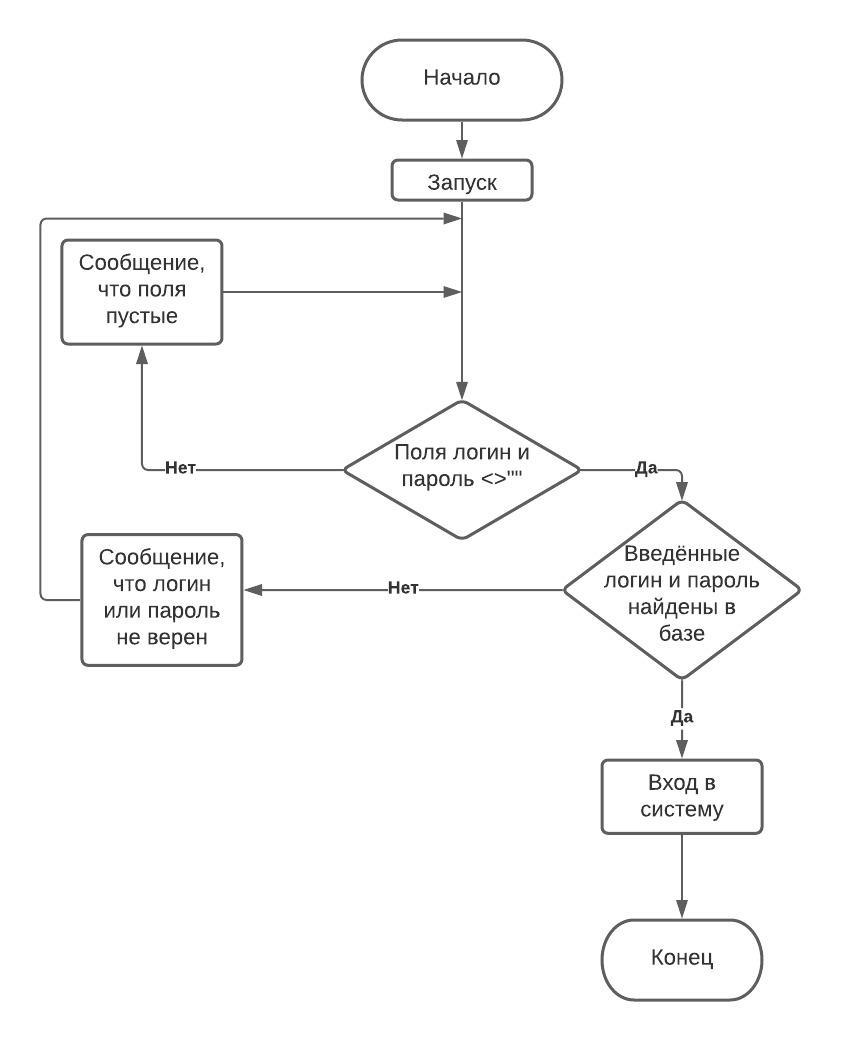
1. Импорт библиотеки
2. Создание главного окна
3. Создание виджет
4. Установка их свойств
5. Определение событий
6. Определение обработчиков событий
7. Расположение виджет на главном окне
8. Отображение главного окна

Более подробно мы разберём этапы создания графического интерфейса в разделе описания проекта.

# Схема работы проекта

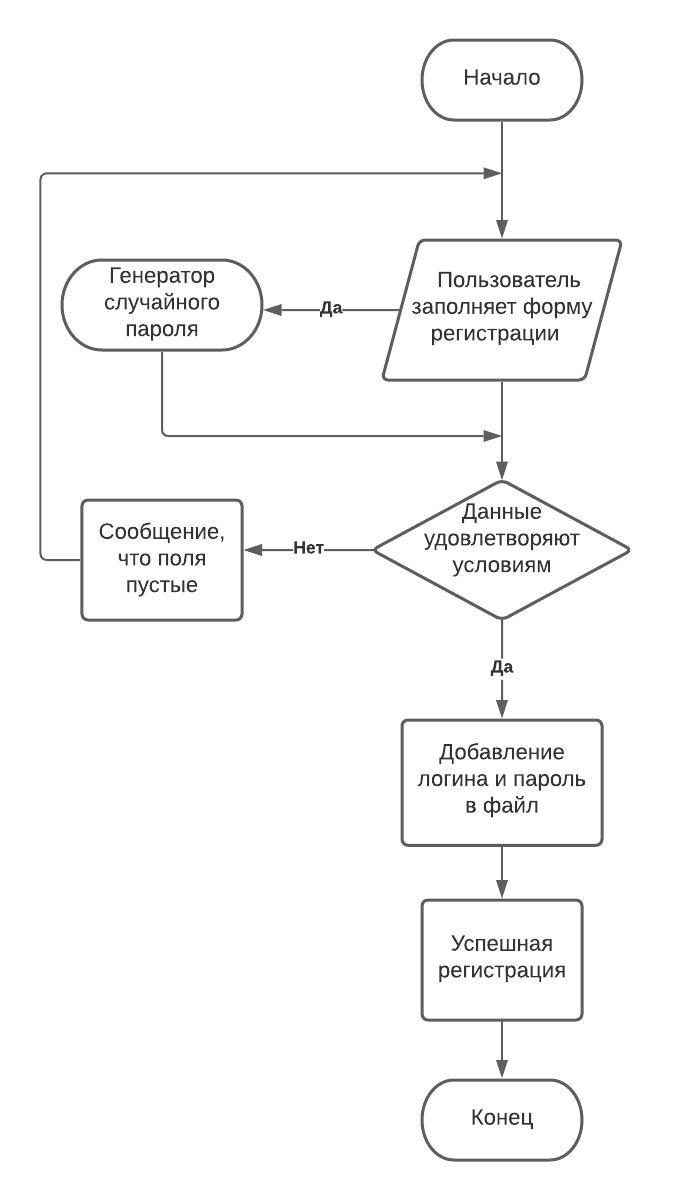
## Cхема авторизации в сети

Таблица 1"Схема авторизации в сети"



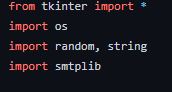
## Схема регистрации в сети

Таблица 2"Cхема авторизации в сети"



# Описание процесса создания проекта

Процесс создания проекта начинается с того, что мы импортируем модуль графического интерфейса Tkinter. Как и любой модуль, tkinter в Python можно импортировать двумя способами: командами import tkinter или from tkinter import \*. В дальнейшем мы будем пользоваться только вторым способом, т. к. это позволит не указывать каждый раз имя модуля при обращении к объектам, которые в нем содержатся.



Затем задаём параметры первого окна. В первом окне у нас будут располагаться кнопки выбора действий: авторизация и регистрация.

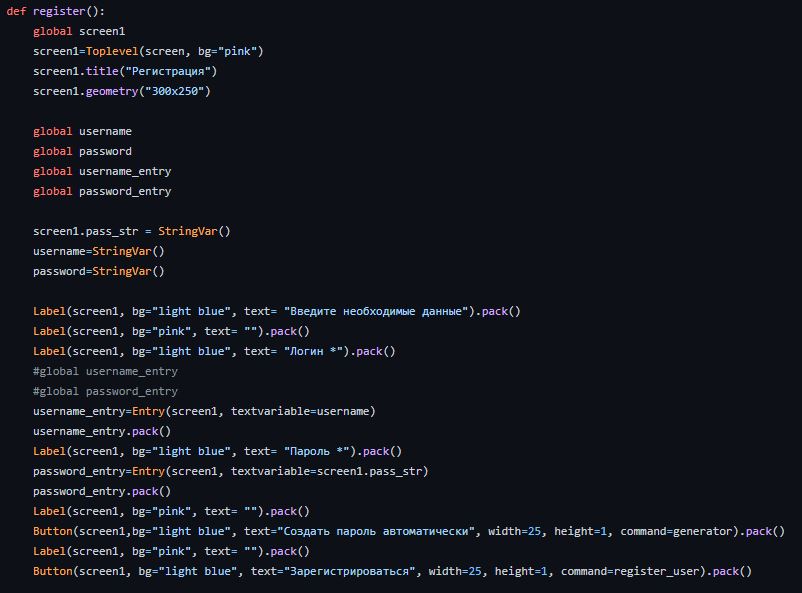
Все необходимые параметры, надписи и кнопки мы объединяем в функцию def main\_screen().



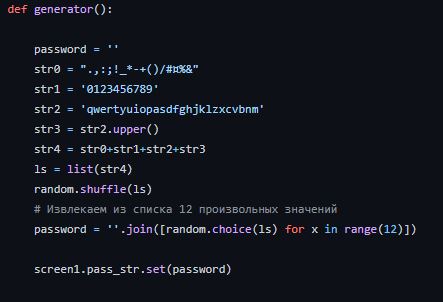
Затем нам необходимо создать команды для кнопок «Регистрация» и «Авторизация».

Для создания команды «Регистрация» мы создаём функцию def register(), а для авторизации def login().

Функция регистрации у нас будет открывать новое окно, в котором пользователь сможет совершить необходимую процедуру регистрации.



Во всплывающем окне пользователю будет необходимо ввести желаемый логин и пароль. Также у пользователя будет возможность сгенерировать пароль случайным образом. Для генерации пароля будет создана отдельная функция def generator () для реализации которой мы импортировали модуль random.



Так как функция def register() открывается у нас в новом окне, то при её создании мы указываем расположение полей ввода, надписей и кнопок на втором экране под названием screen1.

Затем нам необходимо составить список действий, которые будут происходить после того, как пользователь заполнит необходимые поля для регистрации. Введённые пользователем данные у нас будут сохраняться в файл. Для каждого зарегистрированного пользователя у нас будет создаваться отдельный файл с его логином и паролем. Функция добавления данных пользователя в «регистр» у нас будет называться def register\_user(). В случае успешной регистрации заполненные пользователем поля будут автоматически очищаться.



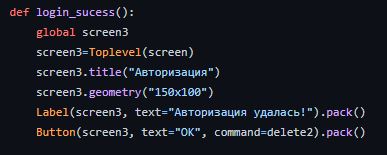
Следующим шагом создания нашего проекта станет функция авторизации def login(). После вызова этой функции у нас будет снова открываться новое окно, которому мы присвоим значение screen2.



В процессе авторизации нам необходимо проверить совпадают ли логин и пароль, введённые пользователем с данными в регистре. Для данной проверки мы создаём функцию def login\_verify(). В обязанности данной функции будет входить считывание информации из созданных файлов. Для считывания файлов из директории проекта мы импортируем модуль OS.

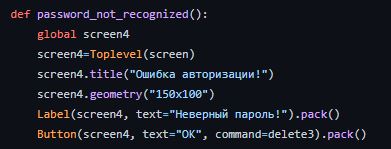


В случае совпадения введённых данных пользователь увидит надпись «Авторизация удалась». Данная информация откроется для пользователя в новом окне, которому мы присвоили значение screen3 и создали в нём функцию def login\_sucess().

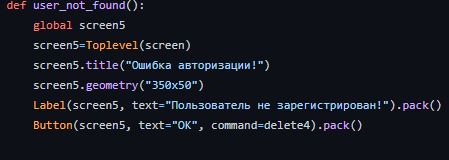


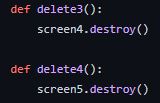
В данную функцию мы добавили кнопку «Ок» нажав на которую программа открывает доступ к приложению для отправки писем.

Если в случае авторизации пользователь введёт неверный пароль, то откроется окно, которому мы присвоили значение screen4 и в нём будет отображена информация о том, что пароль неверный. Нажав на кнопку «Ок» данное окно закроется, и пользователь вернётся к окну авторизации (def delete3()). Функцию ввода неверного пароля мы назвали def password\_recognized().



В случае ввода пользователем неверного логина откроется окно, которому мы присвоили значение screen5 и пользователь будет проинформирован о том, что такой пользователь не зарегистрирован. Нажав на кнопку «Ок» данное окно закроется, и пользователь вернётся к окну авторизации (def delete4()). Функцию ввода неверного логина мы назвали def user\_not\_found().

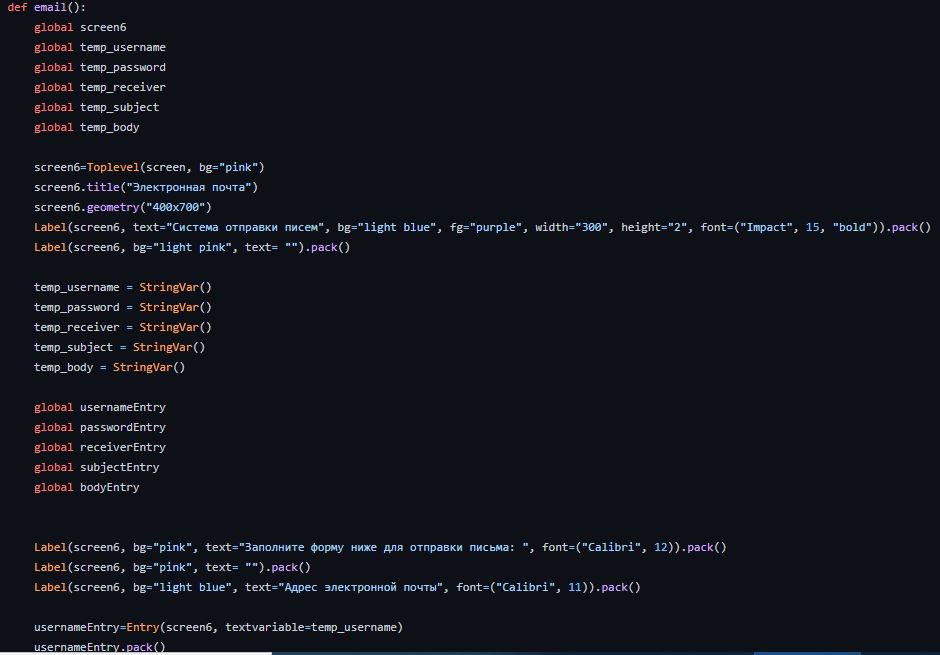


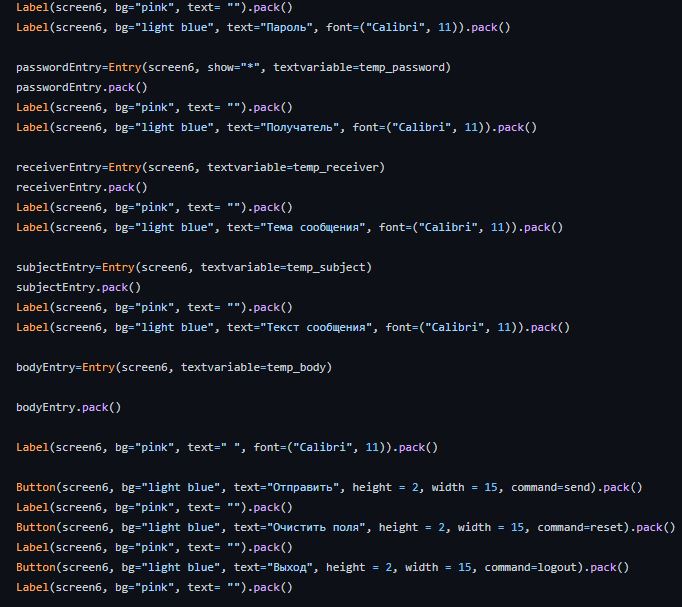


Теперь мы вернёмся к итогу успешного прохождения авторизации. После прохождения авторизации пользователю откроется окно, которому мы присвоили название screen6. Открывшееся окно предоставит нам возможность отправить письмо из аккаунта gmail. Для реализации данной функции нами был импортирован модуль smtplib. Simple Mail Transfer Protocol (SMTP) — это протокол, который обрабатывает отправку электронной почты и маршрутизацию электронной почты между почтовыми серверами.

Python предоставляет модуль smtplib, который определяет объект сеанса клиента SMTP, который может быть использован для отправки почты на любую интернет-машину с помощью SMTP или ESMTP слушателя daemon.

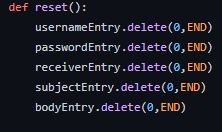
Реализацию данной системы мы начинаем с создания нового окна, которому мы присвоили название screen6. Для создания окна мы создаём функцию def email().





В рамках данной функции мы задаём команды для трёх основных функций: отправка письма (def send()), очищение всех заполненных полей (def reset()), и выход из системы (def logout()).

Функция очистки всех заполненных полей (def reset()) выглядет следующим образом:



В функции отправки писем мы используем возможности модуля smtplib:



После успешной отправки письма у пользователя есть возможность отправить новое письмо или выйти из программы. Для этого мы создали функцию def logout(), при активации которой закроются все окна программы.

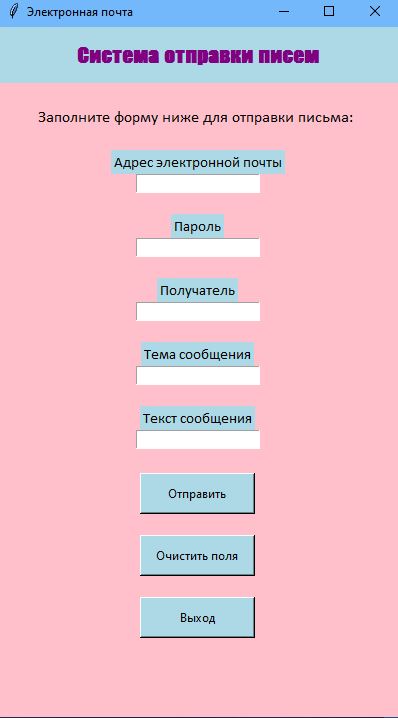


# Краткое описание использования проекта

Созданный проект можно использовать как начало любого более масштабного проекта, для доступа к которому, пользователю будет необходимо авторизироваться. Созданную систему можно будет использовать, например, как вход в личный кабинет, окно которого будет открываться после успешного прохождения регистрации.

В нашем случае созданный проект используется для входа в систему отправки писем.





# Заключение

В ходе создания данного проекта мною были изучены возможности графического интерфейса Tkinter, а также разобрана подробно система функционирования системы регистрации и авторизации пользователя с помощью придуманных или случайно генерированных данных.

В генерации случайных данных нам помог модуль random. Модуль random предоставляет функции для генерации случайных чисел, букв, случайного выбора элементов последовательности.

В процессе создания проекта мы также познакомились с модулем OS, модулем чтения информации из файла, находящегося в директории. Модуль OS предоставляет множество функций для работы с операционной системой.

В качестве вывода из данного проекта можно привести его конечный результат, а точнее нашу систему отправки писем, доступ к которой был получен путём прохождения операций регистрации и авторизации в системе. Для получения данного результата мы прибегли к модулю smtplib.

У модуля графического интерфейса Tkinter имеется огромная библиотека дополнительных модулей, с помощью которых мы можем создать неограниченное количество решений для реализации конечной цели созданного нами проекта.

# Список используемой литературы

Свободная энциклопедия Википедия, URL [www.ru.wikipedia.org](http://www.ru.wikipedia.org)

Официальный сайт языка программирования Python, URL [www.python.org](http://www.python.org)

Всемирная поисковая система Google, URL [www.google.com](http://www.google.com)

Курс по библиотеке Tkinter языка Python, URL [www.wikiversity.org](http://www.wikiversity.org)

# Содержание

Введение 3

I. Схема работы проекта 5

A. Cхема авторизации в сети 5

B. Схема регистрации в сети 6

II. Описание процесса создания проекта 7

III. Краткое описание использования проекта 16

Заключение 17

Список используемой литературы 18

Содержание 19