**ДЕПАРТАМЕНТ ОБРАЗОВАНИ И НАУКИ ГОРОДА МОСКВЫ**

**Северо-Восточный административный округ**

**Государственное бюджетное профессиональное**

**образовательное учреждение города Москвы**

**«Политехнический колледж имени П.А. Овчинникова»**

**Специальность: 09.02.07 «Информационные системы и программирование»**

**Курсовая работа**

**По МДК 01.01**

**Создание мессенджера**

**на языке программирования Python**

Выполнила студентка 2 курса

группы 2ИС Комаеда Н.Ю.

Руководитель Олейник В.И.

МОСКВА 2023

**Оглавление**

Введение 4

Разработка мессенджера 5

Анализ существующих решений 5

Суть приложения 5

Состав клиента 6

Состав сервера 7

Составление требований к программному продукту 8

Введение 8

Наименование программного продукта 8

Основание для разработки 8

Назначение разработки 8

Область применения 8

Требования к программному продукту 8

Требования к функциональным характеристикам 8

Спецификация качества программного продукта 9

Практичность 9

Функциональность 9

Требования к надежности 9

Требования к обеспечению надежного функционирования программного продукта 9

Время восстановления после отказа 10

Отказы из-за некорректных действий оператора 10

Условия эксплуатации 10

Климатические условия эксплуатации 10

Требования к квалификации и численности персонала 10

Требования к информационной и программной совместимости 11

Требования к информационным структурам 11

Требования языкам программирования 11

Требования к программным средствам, используемым программой 11

Требования к защите информации 11

Требования к документации 11

Стадии и этапы разработки 11

Стадии разработки 11

Этапы разработки 12

Содержание работ по этапам 12

Анализ требований 12

Проектирование архитектуры и интерфейсов 12

Проектирование архитектуры 12

Проектирование интерфейсов 13

Разработка 26

Установка Python 26

Установка необходимых библиотек 28

Описание алгоритма на естественном языке 28

Введение

*«В начале было Слово…»*

*От Иоанна 1:1*

Общение запустило двигатель прогресса: объединило первые племена, основало первые цивилизации, совершило первые открытия, и, в конце концов, привело нас к сегодняшнему дню. Общение – это неотъемлемая часть человеческой сущности. По мере развития общества появлялись новые и новые средства коммуникации: сначала, как предполагают ученые, это были жесты и звуки, потом появилась речь, затем письмо. В последние сто лет технологии развиваются экспоненциально, и вместе с новыми открытиями появились и новые возможности.

Возможность обмениваться информацией мгновенно стала неотъемлемой частью жизни современного человека, в том числе и меня, поэтому я решила выбрать написание мессенджера на языке программирования Python в качестве темы для моей курсовой работы.

Мессенджеры позволяют лучше планировать и координировать действия на работе, помогают получать важные новости, оставаться на связи с семьей и друзьями в реальном времени даже на большом расстоянии.

Язык программирования Python предлагает широкий набор библиотек, которые облегчают процесс разработки. К тому же, функции большинство команд интуитивно понятны, ведь многие из них это аббревиатуры, словосочетания или слова на английском языке.

Разработка мессенджера

# Анализ существующих решений

1. Для работы с языком программирования Python существует множество ПО: Visual Studio, Notepad++, Spyder, Thonny, PyCharm (и так далее). Для разработки мессенджера я буду работать с PyCharm, так как у меня есть опыт работы с ним, он интуитивен в использовании и имеет широкий функционал
2. Для хранения информации понадобится база данных. Для этой цели я буду использовать Microsoft Excel. Для работы с данным ПО я буду пользоваться библиотекой openpyxl
3. Для обеспечения коммуникации между клиентом и сервером я буду использовать модуль socket. Также мне нужно обеспечить многозадачность, для чего я буду использовать модуль threading
4. Для создания окон я буду использовать модуль tkinter из встроенной библиотеки Python
5. В работе приложения мне также нужно будет работать с файлами, для чего я буду использовать библиотеки os и glob
6. Для тестирования программы понадобится pytest
7. Для облегчения процесса разработки я буду использовать GitHub

# Суть приложения

Мессенджер должен обеспечивать следующие возможности:

1. Регистрация, входа в аккаунт
2. Изменение фотографии профиля из предложенного набора картинок, изменение юзернейма и пароля, выход из аккаунта
3. Поиск пользователей по юзернейму, просмотр контактов
4. Отправка сообщений, обновление истории в реальном времени, очистка истории
5. Закрытие программы

Сервер должен позволять работать с несколькими клиентами сразу, обеспечивать выполнение нужных операций в зависимости от полученных от клиента запросов.

Мессенджер состоит из двух приложений: клиент и сервер

# Состав клиента

1. Окно входа в аккаунт:   
   поля ввода «логин» и «пароль»;   
   кнопки «войти» и «зарегистрироваться»
2. Окно регистрации:   
   поля ввода «юзернейм», «логин», «пароль», «повторите пароль»;   
   кнопки «зарегистрироваться» и «🡨» (назад)
3. Главное окно:   
   кнопки «настройки», «друзья», «сообщения», «помощь» (при нажатии открывается окно с информацией), «закрыть»
4. Страница настроек:   
   кнопки «изменить юзернейм» (при нажатии появляется поле ввода), «изменить фото» (при нажатии открывается окно с доступными картинками), «изменить пароль» (при нажатии открывается окно с полем ввода и кнопкой «сохранить изменения»), «выйти из аккаунта», «главное меню»
5. Страница друзей:   
   поле ввода юзернейма и кнопка «найти»;   
   кнопки с юзернеймами и фотографиями друзей (появляются в случае если у пользователя были найдены контакты, при нажатии переходит на страницу «сообщения», открывая диалог с выбранным пользователем);   
   кнопка «отправить сообщение» (появляется в случае если пользователь был найден);   
   кнопки «🡨» (назад) и «🡪» (вперед) для просмотра списка друзей (появляются в случае если количество друзей превышает 4);   
   кнопки «открыть избранное» (открывает переписку с самим собой) и «главное меню»
6. Страница сообщений:   
   содержит два текстовых поля, одно из которых – история переписки (нельзя редактировать), а другое предназначено для ввода сообщений. Отправка происходит по нажатию на enter;  
   кнопка «очистить»

# Состав сервера

1. Запуск сервера, обеспечение многопоточности
2. Работа с базой данных
3. Прием запросов
4. Отправление запросов
5. Обработка запросов от клиента

Составление требований к программному продукту

**(ТЕХНИЧЕСКОЕ ЗАДАНИЕ)**

# Введение

## Наименование программного продукта

Наименование: checkmate

## Основание для разработки

Выполнение курсовой работы, углубление знаний в языке программирования Python

## **Назначение разработки**

Цель проекта – разработка мессенджера, который представляет собой небольшое экранное приложение. Он позволит пользователям обмениваться сообщениями в режиме реального времени. Проект планируется совершенствовать в будущем.

## Область применения

Клиент мессенджера предназначен для обеспечения удобной коммуникации между пользователями. Предназначен для широкой аудитории пользователей всех возрастных групп; предполагается, что он будет использоваться для личных целей.  
Сервер мессенджера контролируется Заказчиком.

# Требования к программному продукту

## Требования к функциональным характеристикам

Сервер должен обеспечивать выполнение перечисленных ниже функций:

1. Запуск сервера
2. Печать отчета о каждом запросе с клиента в консоль
3. Прием и отправка запросов, обеспечение целостности данных
4. Выполнение определенных действий в соответствии с полученными запросами
5. Многозадачность: работа с каждым подключенным клиентом по отдельности
6. Изменение и чтение базы данных
7. Запись историй сообщений пользователей в текстовые файлы
8. Обработка ошибок

Клиент должен обеспечивать выполнение перечисленных ниже функций:

1. Подключение к серверу
2. Прием и отправление запросов, обеспечение целостности данных
3. Выполнение определенных действий в соответствии с полученными запросами
4. Вход в аккаунт, выход из аккаунта; автоматический вход в аккаунт при запуске приложения, если выход выполнен не был
5. Регистрация пользователей
6. Изменение фотографии профиля
7. Изменение юзернейма и пароля
8. Поиск пользователей
9. Просмотр контактов
10. Отправление сообщений другим пользователям
11. Обновление истории переписки в реальном времени
12. Очистка истории сообщений
13. Закрытие программы
14. Обработка ошибок

# Спецификация качества программного продукта

## Практичность

Клиент программы должен иметь понятный пользовательский графический интерфейс и небольшой размер

## Функциональность

Программный продукт должен являться полностью завершенным программным обеспечением.

# Требования к надежности

## Требования к обеспечению надежного функционирования программного продукта

Надежное (устойчивое) функционирование программы должно быть обеспечено выполнением Заказчиком совокупности организационно-технических мероприятий, перечень которых приведен ниже:

а) организацией бесперебойного питания технических средств;

б) использованием лицензионного программного обеспечения;

в) регулярным выполнением рекомендаций Министерства труда и социального развития РФ

г) регулярным выполнением требований ГОСТ 51188-98. Защита информации. Испытания программных средств на наличие компьютерных вирусов

## Время восстановления после отказа

Время восстановления после отказа, вызванного сбоем электропитания технических средств (иными внешними факторами), не фатальным сбоем (не крахом) операционной системы, не должно превышать 30-ти минут при условии соблюдения условий эксплуатации технических и программных средств.

Время восстановления после отказа, вызванного неисправностью технических средств, фатальным сбоем (крахом) операционной системы, не должно превышать времени, требуемого на устранение неисправностей технических средств и переустановки программных средств.

## Отказы из-за некорректных действий оператора

Отказы клиента возможны вследствие некорректных действий оператора (пользователя) при взаимодействии с операционной системой.

Во избежание возникновения отказов клиента по указанной выше причине следует обеспечить работу конечного пользователя без предоставления ему административных привилегий.

# Условия эксплуатации

## Климатические условия эксплуатации

Климатические условия эксплуатации, при которых должны обеспечиваться заданные характеристики, должны удовлетворять требованиям, предъявляемым к техническим средствам в части условий их эксплуатации

## Требования к квалификации и численности персонала

Минимальное количество персонала, требуемого для работы клиента – один человек (оператор).

Минимальное количество персонала, требуемого для работы сервера – один человек (системный администратор).

Системный администратор должен знать язык программирования Python и основы компьютерных сетей.

Задачи системного администратора:

а) поддержание работоспособности технических средств;

б) установка и поддержание работоспособности системных программных средств — операционной системы;

в) установка программного продукта;

г) создание резервных копий базы данных;

# Требования к информационной и программной совместимости

## Требования к информационным структурам

Для создания базы данных используется Microsoft Excel. Необходимо обеспечить серверу возможность ее изменять.

## Требования языкам программирования

Программный продукт должен быть написан на языке программирования Python.

## Требования к программным средствам, используемым программой

Системные программные средства, используемые программным продуктом, должны быть представлены версией операционной системы Windows 7 или выше и Microsoft Excel.

## Требования к защите информации

Необходимо ограничить доступ пользователей к базе данных и текстовым файлам с историями сообщений, не принадлежащим им.

# Требования к документации

Документация прилагается

# Стадии и этапы разработки

## Стадии разработки

Разработка должна включать в себя следующие этапы:

1. Планирование
2. Анализ
3. Проектирование
4. Кодирование
5. Тестирование
6. Внедрение
7. Сопровождение

## Этапы разработки

1. Анализ требований
2. Проектирование архитектуры и интерфейсов
3. Разработка
4. Тестирование клиента и сервера по отдельности
5. Тестирование взаимодействия клиента и сервера
6. Документирование и выпуск

Содержание работ по этапам

# Анализ требований

Сбор информации, создание, анализ и проверка требований

# Проектирование архитектуры и интерфейсов

## Проектирование архитектуры

Для разработки мессенджера была выбрана клиент-серверная архитектура. В данной модели есть два основных компонента: клиент, который работает на устройстве пользователя, и сервер, который обрабатывает запросы от клиентов.

Преимущества клиент-серверной архитектуры  
Клиенты отвечают только за пользовательский интерфейс и взаимодействие с пользователем. Сервер же обрабатывает принятые от клиента запросы и хранит данные.

Обоснование выбора архитектуры  
Хранение данных на сервере позволяет обеспечить надежную работу с информацией пользователей.

Разделение компонентов  
А) Клиент

Отвечает за пользовательский интерфейс и взаимодействие с пользователем  
Отправляет запросы на сервер и получает ответы с сервера

Б) Сервер

Обрабатывает запросы, полученные от клиента  
Читает и изменяет базу данных

В) База данных

Хранит информацию о пользователях и истории переписок

## Проектирование интерфейсов

А) Клиент

При запуске программы в первый раз появляется окно входа в аккаунт (Рис. 1.1)

Данное окно содержит поля ввода «логин» и «пароль», куда предлагается ввести соответствующие данные если пользователь уже зарегистрировался. В целях безопасности вместо вводимых символов в поле ввода «пароль» пользователь видит звездочки.

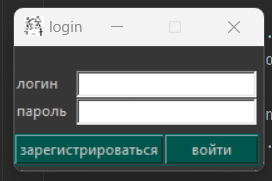
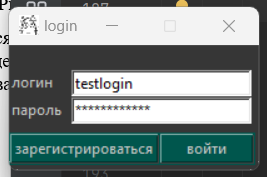
 

Рис. 1.1 «Окно входа в аккаунт»  
  
Если пользователь ввел в поле ввода текст, содержащий символ «:», и нажал на кнопку «войти», на экран выведется сообщение «недопустимые символы (:)» (Рис 1.2). Это сделано затем, чтобы предотвратить ошибку: сервер принимает запрос от клиента и разделяет полученное сообщение по знаку «:» (login:password).

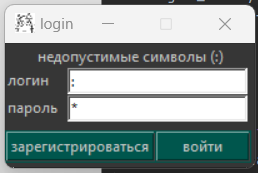


Рис 1.2 «недопустимые символы (:)»

Если пользователь ввел в поле логина «login» или в поле пароля «password» и нажал на кнопку «войти», на экран выведется сообщение «не-а» (Рис 1.3). Это сделано затем, чтобы исключить ошибку при чтении базы данных, так как первая строка таблицы содержит колонки именно с такими названиями.

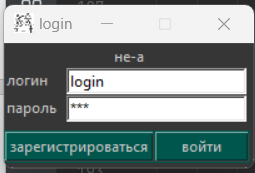
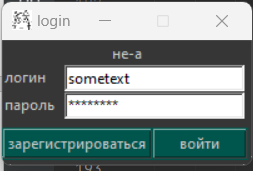
 

Рис 1.3 «не-а» (на втором скриншоте в поле «пароль» введено «password»)  
  
Если введенные пользователем логин и пароль не совпадают с информацией в базе данных, после нажатия на кнопку «войти», на экран выведется сообщение «>\_< некорректные данные» (Рис. 1.4)

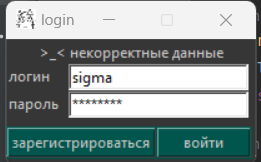


Рис 1.4 «некорректные данные»

Если пользователь еще не зарегистрирован, то ему предлагается создать аккаунт. Для этого нужно нажать на кнопку «зарегистрироваться», после чего появляется окно регистрации (Рис 2.1) с полями «юзернейм» (то, что будет отображаться в профиле), «логин» (с помощью него будет производиться вход в аккаунт), «пароль» и «повторите пароль» (нужно для уверенности в том, что пользователь ввел желаемый пароль корректно). Также слева есть кнопка «🡨», по нажатию на которую пользователя возвращает на окно входа в аккаунт.

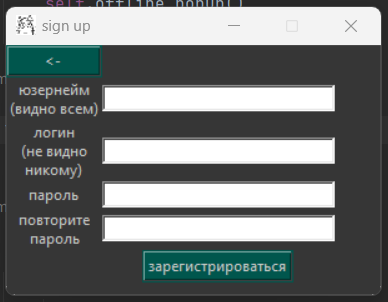


Рис. 2.1 «Окно регистрации»  
  
Если хотя бы одно из полей пустое, на экран выведется сообщение «заполните все поля» после нажатия на кнопку «зарегистрироваться» (Рис. 2.2).

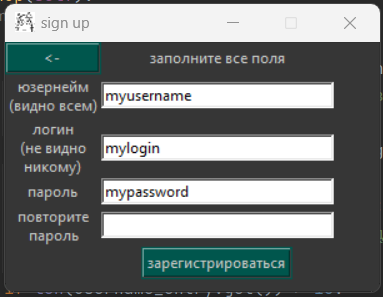


Рис. 2.2 «заполните все поля»  
  
Если пользователь ввел хотя бы в одно из полей текст, содержащий символ «:», после нажатия на кнопку «зарегистрироваться» на экран выведется сообщение «недопустимые символы (:)» (Рис 2.3)

Если пользователь ввел в поле «юзернейм» текст, содержащий более 10 символов, после нажатия на кнопку «зарегистрироваться» на экран выведется сообщение «слишком длинный юзернейм», и содержимое соответствующего поля будет обрезано до 10 символов (Рис 2.4). Это нужно для того, чтобы юзернейм правильно отображался.

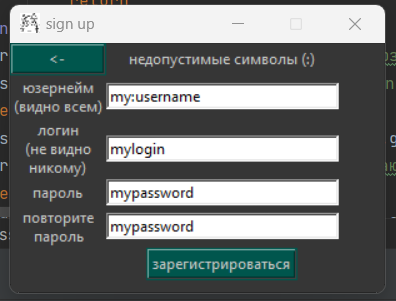


Рис 2.3 «недопустимые символы (:)»

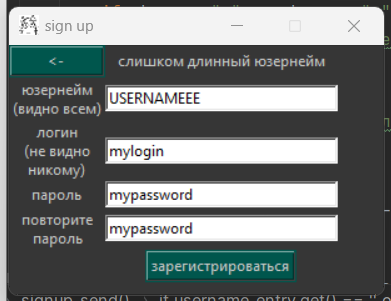
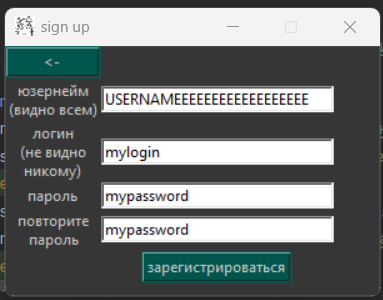


Рис 2.4 «слишком длинный юзернейм»

Если данные, введенные в поле «пароль» не совпадают с данными, введенными в поле «повторите пароль», после нажатия на кнопку «зарегистрироваться» на экран выведется сообщение «пароли не совпадают!» (Рис 2.5)

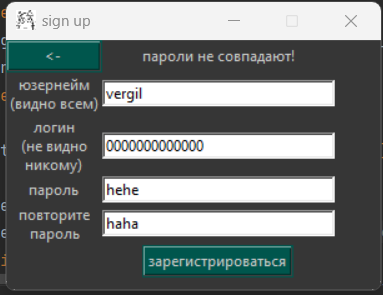


Рис 2.5 «пароли не совпадают!»

Если пользователь ввел в поле логина «login» или в поле пароля «password», после нажатия на кнопку «зарегистрироваться» на экран выведется сообщение «не-а» (Рис 2.6.). Это сделано затем, чтобы предотвратить ошибку при чтении базы данных, поскольку первый ряд таблицы содержит колонки именно с такими названиями.

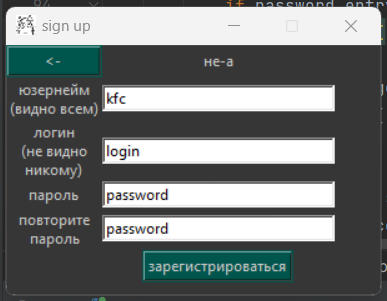


Рис 2.6 «не-а»

При успешной регистрации после нажатия на кнопку «зарегистрироваться» выводится сообщение с информацией, введенной пользователем, и открывается главное меню мессенджера (Рис. 2.7, Рис 4)

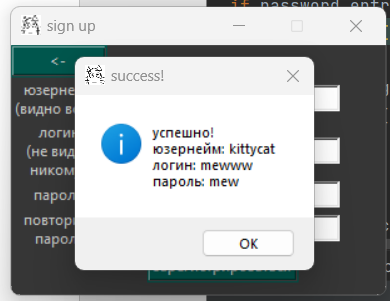


Рис 2.7 «успешная регистрация»  
  
 C:\Users\ajngc\AppData\Local\Microsoft\Windows\INetCache\Content.Word\profile0.pngC:\Users\ajngc\AppData\Local\Microsoft\Windows\INetCache\Content.Word\profile1.pngC:\Users\ajngc\AppData\Local\Microsoft\Windows\INetCache\Content.Word\profile2.pngC:\Users\ajngc\AppData\Local\Microsoft\Windows\INetCache\Content.Word\profile3.pngC:\Users\ajngc\AppData\Local\Microsoft\Windows\INetCache\Content.Word\profile4.png

Рис 3.2 «Фотографии профилей»



Рис 3.1 «Иконка приложения»

Главное меню содержит в себе кнопки «настройки», «друзья», «сообщения», «помощь», «закрыть» (Рис. 4)

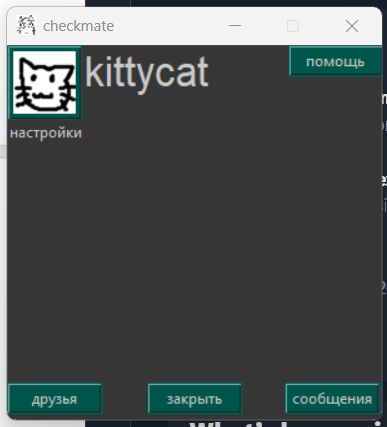


Рис. 4 «Главное меню»

При нажатии на кнопку «настройки» открывается окно настроек, содержащее кнопки «изменить юзернейм», «изменить фото», «изменить пароль», «выйти из аккаунта», «друзья», «сообщения», «главное меню» (Рис. 5.1). При повторном нажатии на кнопку настроек также открывается главное меню. При нажатии на кнопку «выйти из аккаунта» производится выход в окно входа в аккаунт.

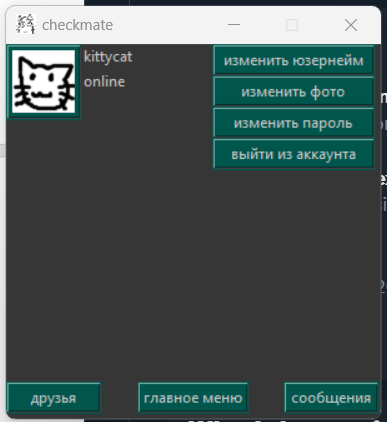


Рис. 5.1 «Окно настроек»

При нажатии на кнопку «изменить юзернейм» на месте юзернейма появляется поле ввода, в которое предлагается ввести новый (Рис. 5.2.1). По нажатию на кнопку «сохранить изменения» юзернейм обновляется.

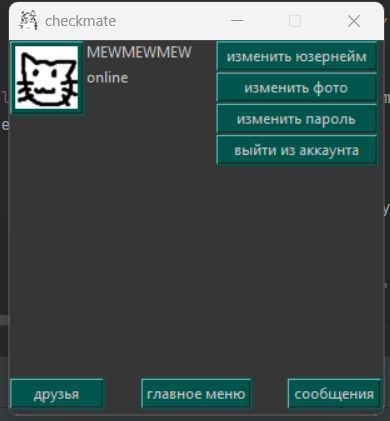
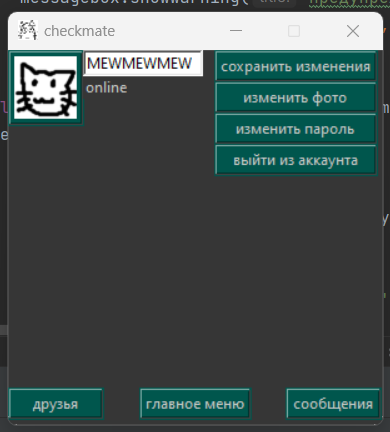
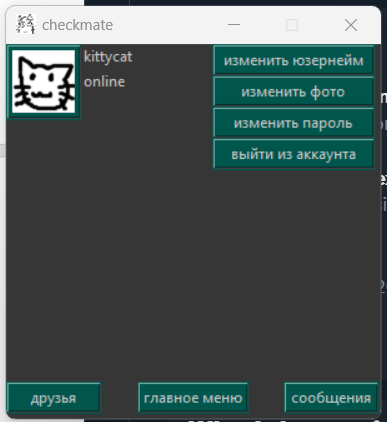


Рис. 5.2.1 «Изменение юзернейма»

Если длина введенного юзернейма превышает 10 символов, после нажатия на кнопку «сохранить изменения» появляется окно с сообщением «слишком длинный юзернейм! максимальная длина – 10 символов». Содержимое поля ввода обрезается до 10 символов (Рис. 5.2.2) Это нужно для того, чтобы юзернейм правильно отображался.

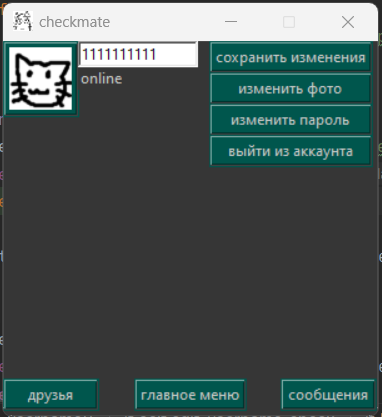
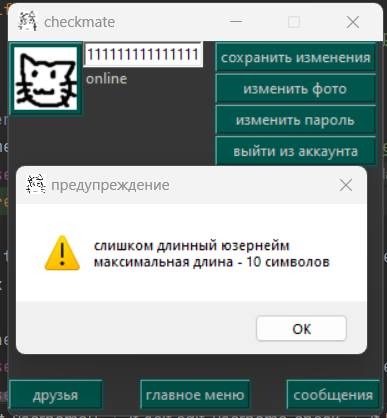


Рис. 5.2.2 «слишком длинный юзернейм»

Если поле ввода пустое, после нажатия на кнопку «сохранить изменения» появляется окно с сообщением «юзернейм не может быть пустым» (Рис. 5.2.3)

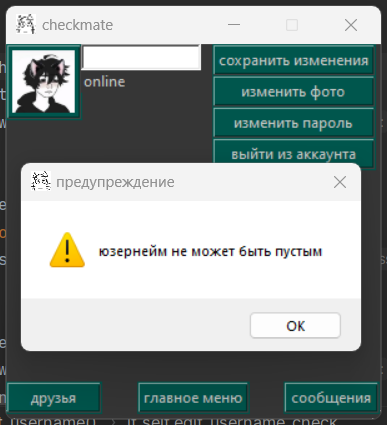


Рис. 5.2.3 «юзернейм не может быть пустым»

Если пользователь ввел в поле текст, содержащий символ «:», появляется окно с сообщением «недопустимые символы (:)» (Рис. 5.2.4)

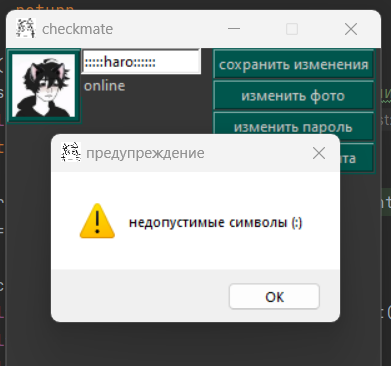


Рис. 5.2.4 «недопустимые символы (:)»

Если введенный пользователем юзернейм уже занят (или является словом «username»), появляется окно с сообщением «вы не можете использовать этот юзернейм:(» (Рис. 5.2.5)

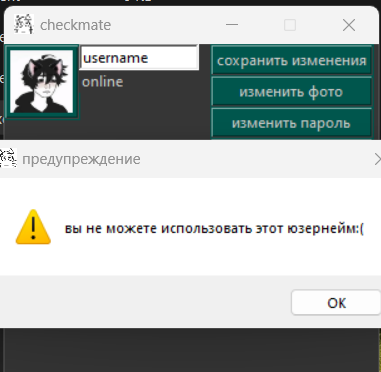
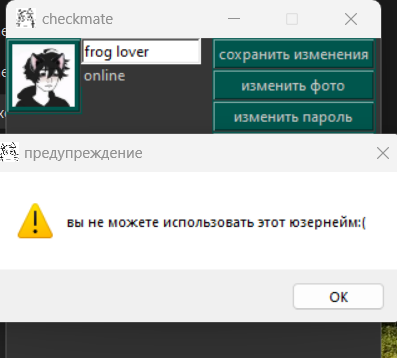


Рис. 5.2.5 «вы не можете использовать этот юзернейм:(»

При нажатии на кнопку «изменить фото» открывается окно с пятью кнопками, которые представляют собой разные картинки. По нажатию на кнопку фотография профиля будет изменена на соответствующую (Рис 5.3). Подробнее см. Рис. 3.2

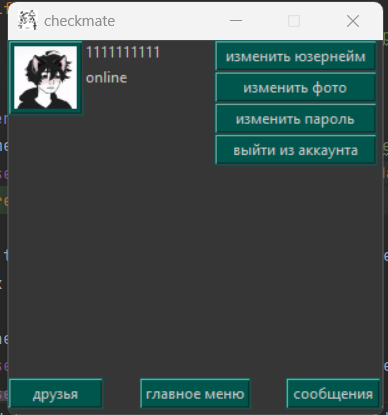
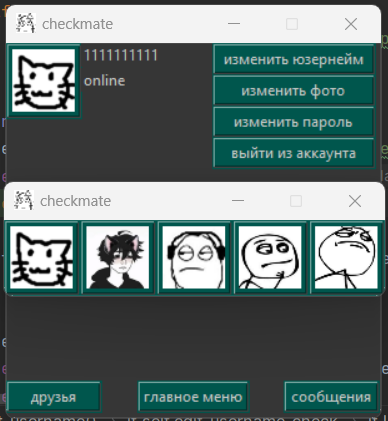


Рис. 5.3 «Изменение фотографии профиля»

При нажатии на кнопку «изменить пароль» открывается окно изменения пароля с полем ввода и кнопкой «сохранить изменения» (Рис. 5.4.1)

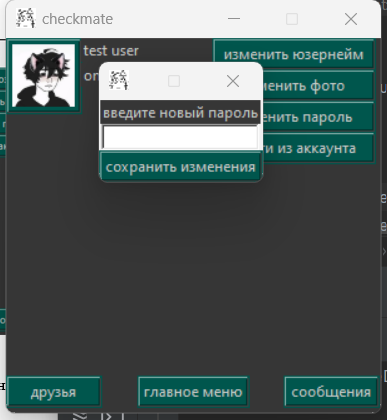


Рис. 5.4.1 «окно изменения пароля»

Если пользователь оставил поле пустым, появляется сообщение «пароль не может быть пустым». Если пользователь ввел «password» - «вы не можете использовать этот пароль». Если поле содержит символ «:» - «недопустимые символы (:)» (Рис. 5.4.2)

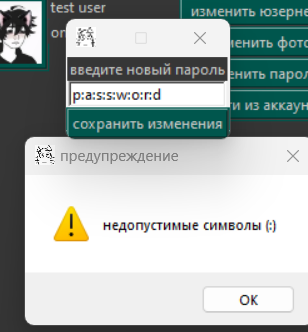
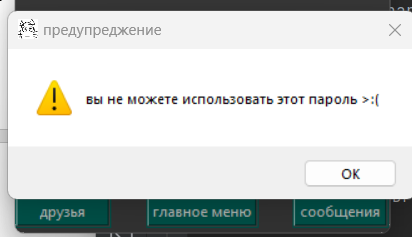
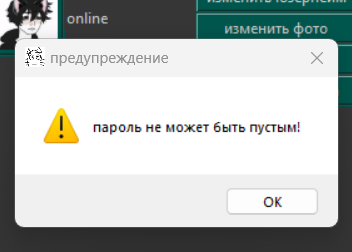


Рис. 5.4.2 «Сообщения при некорректном вводе нового пароля»

При нажатии на кнопку «друзья» открывается страница контактов (Рис. 6.1)

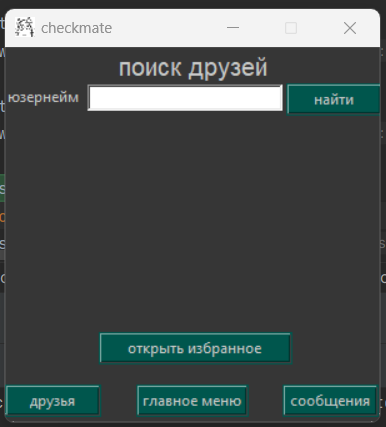


Рис. 6.1. «Страница контактов»

После нажатия на кнопку «найти» производится поиск пользователей в соответствии с данными, введенными в поле. Если пользователь не был найден, выведется «такой пользователь не существует». Если введенные данные совпадают с юзернеймом владельца аккаунта, выведется «вы нашли себя!». Если же пользователь найден был, выведется «пользователь найден», его имя и кнопка «отправить сообщение», при нажатии на которую открывается переписка с этим пользователем (Рис. 6.2)

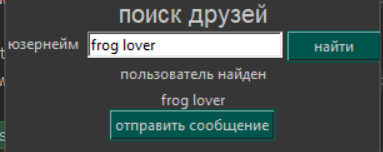
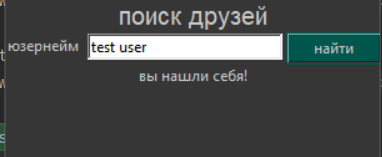
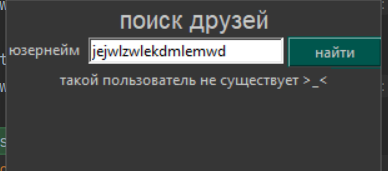
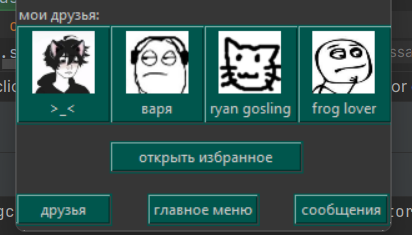
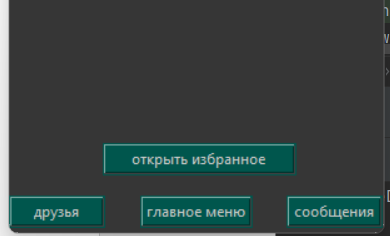
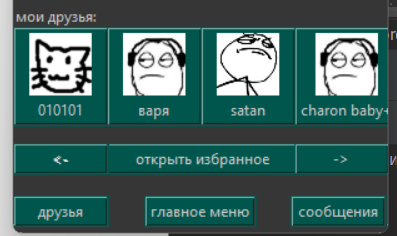


Рис. 6.2. «Поиск друзей»

Если владелец аккаунта еще ни разу не открывал переписку с другим пользователем, список контактов будет пуст. Если друзей меньше 5, будут показываться только они. Если друзей больше 5, появятся кнопки вправо и влево, по нажатию на которые можно листать контакты (Рис. 6.3)



  
Рис 6.3 «мои друзья»

По нажатию на кнопку «открыть избранное» открывается переписка с самим собой.

По нажатию на кнопку «сообщения» открывается последняя открытая переписка с момента открытия приложения. По умолчанию открывается переписка с самим собой. После выхода из приложения последняя переписка сбрасывается до значения по умолчанию. Отправление сообщений производится по нажатию клавиши enter.  
По нажатию на кнопку «очистить» история сообщений очищается, причем очищается у обоих пользователей (Рис 7.1).

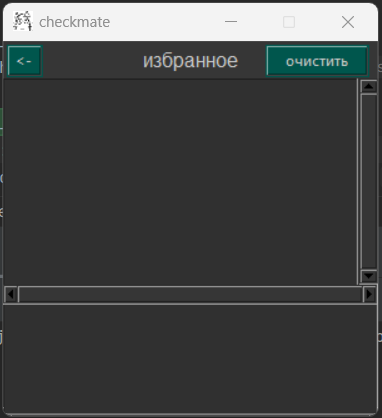
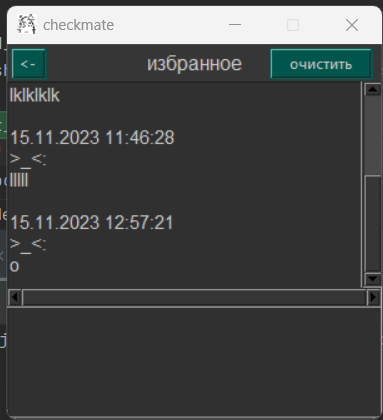


Рис. 7.1 «переписка; очистка истории сообщений»

При нажатии на кнопку «🡨» открывается главное меню.

При нажатии на кнопку «помощь» открывается окно, содержащее адрес электронной почты разработчика и кнопку «скопировать». (Рис. 8) При нажатии на нее адрес копируется в буфер обмена.

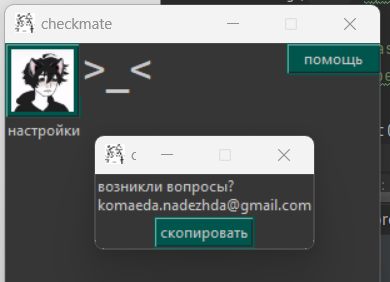


Рис. 8 «помощь»

При нажатии на кнопку «закрыть» приложение закрывается.   
Если выход из аккаунта произведен не был, при повторном открытии приложения произведется автоматический вход в аккаунт и откроется главное меню.

Б) Сервер

Сервер представляет собой консольное приложение, куда выводятся отчеты о полученных от клиента запросах и их обработке. (Рис. 9)

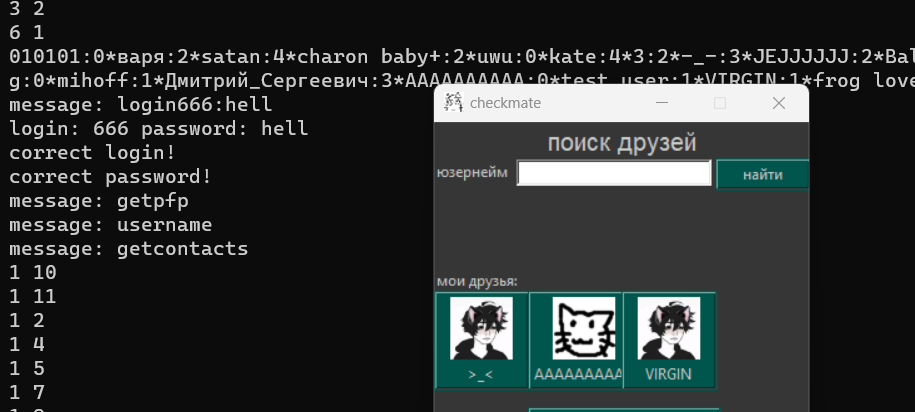


Рис. 9 «Интерфейс сервера»

# Разработка

На данном этапе производится кодирование приложений клиента и сервера и их отладка.

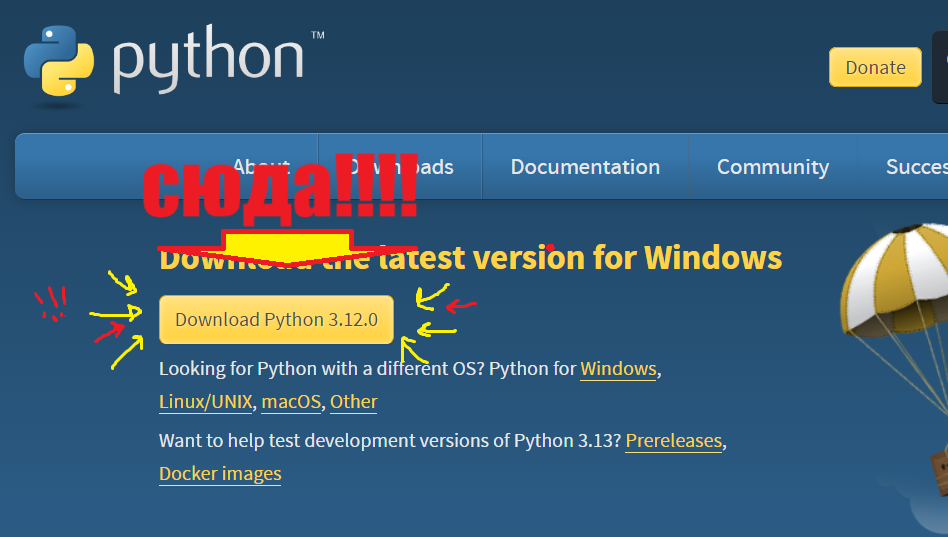
Разработка включает в себя:

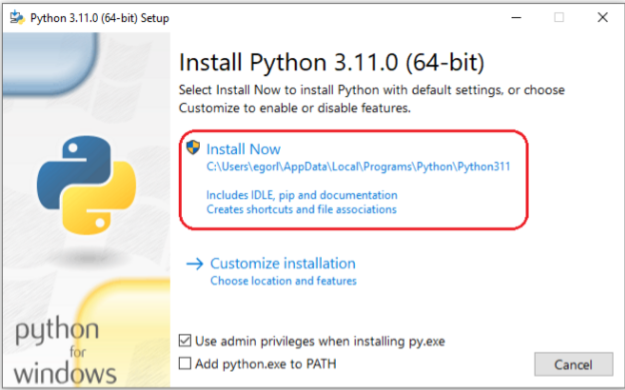
1. Установка Python
2. Установка необходимых библиотек
3. Написание алгоритма на естественном языке
4. Реализация кода программы
5. Листинг кода

## Установка Python

Для работы с языком программирования Python потребуется интерпретатор. Чтобы его установить, нужно перейти на официальную страницу <https://www.python.org/downloads/>, а затем скачать последнюю версию, нажав кнопку «Download Python» (Рис. 10)

После нажатия на кнопку установщик загрузится на ваш компьютер.

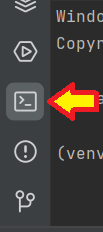


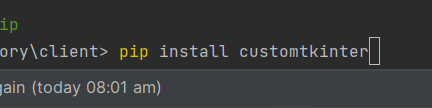
После успешного скачивания интерпретатора и запуска установщика открывается мастер установки. Нажимаем Install Now

После завершения установки можно проверить успешность установки Python и его версию, прописав в командную строку следующую команду: python –version

## Установка необходимых библиотек

Для установки необходимых библиотек достаточно прописать в локальном терминале команду pip install имя\_библиотеки и нажать enter.



1)   
  
  
 2)

## Описание алгоритма на естественном языке

Клиент

Состав класса messenger:

1. Метод подключения к серверу (establish\_connection)

Метод пытается установить соединение с сервером через TCP-сокет. Если подключение успешно, вызывается метод login. Если сервер недоступен, вызывается метод offline\_popup и программа завершается.

1. Метод для получения данных с сервера (receive)

Метод получает данные до тех пор, пока не будет получен символ “|”. Затем производится декодирование в UTF-8 и удаление символа “|”. Если в процессе генерируется исключение, вызывается метод offline\_popup и программа завершается.

1. Метод для отправки данных на сервер (transmit)

Принимает параметр message. Метод пытается отправить сообщение, изменяя его тип данных на str, добавляя символ “|” в конце и кодируя его в UTF-8. Если генерируется исключение, вызывается метод offline\_popup и программа завершается.

1. Метод открытия окна, оповещающего о том, что сервер сейчас оффлайн (offline\_popup)

Создает всплывающее окно с текстом «сервер оффлайн» и кнопкой «закрыть». При нажатии на нее производится выход из приложения.

1. Метод регистрации (signup)

Создает окно с полями ввода и кнопками, также имеет вложенную функцию signup\_send, которая проверяет данные, введенные пользователем. Если все проверки пройдены, данные отправляются на сервер с помощью метода transmit, получается ответ с помощью receive, и в зависимости от ответа выводится сообщение об ошибке или регистрация завершается.   
Также есть еще одна вложенная функция go\_back, которая возвращает в окно входа в аккаунт.

1. Метод входа в аккаунт (login)

Создает окно входа в аккаунт с полями ввода и кнопками «зарегистрироваться» и «войти».  
Содержит вложенную функцию login\_password\_check, которая проверяет введенные пользователем данные. Если все проверки пройдены, на сервер передаются логин и пароль, введенные пользователем. Потом производится получение ответа, и в зависимости от него либо производится вход в аккаунт, либо отображается ошибка.  
Также присутствует вложенная функция redirect, которая открывает окно регистрации.  
instant\_login и instant\_login\_check позволяют автоматически войти в аккаунт, если выход из него произведен не был.

1. Метод закрытия приложения close\_window

Устанавливает значение переменной needs\_updating\_check на False, чтобы остановить поток, обновляющий историю сообщений, и закрывает приложение.

1. Метод main\_info

Внутри этого метода создаются основные виджеты, добавляются картинки и устанавливается тема приложения.   
Также внутри есть функция help, которая создает окно с информацией. Внутри этой функции есть еще одна функция copy, которая копирует электронную почту разработчика в буфер обмена.

1. Метод профиля profile\_click

В этом методе обрабатываются действия, связанные с профилем, такие как изменение юзернейма (edit\_username), изменение пароля (edif\_password), изменение картинки профиля (edit\_pfp), и выход из аккаунта (log\_out). Измененные данные сверяются с базой данных и в случае успеха отправляются на сервер, где вносятся изменения в базу данных

1. Метод контактов contacts\_click

Создает и размещает разные виджеты.

Имеет множество вложенных функций:  
- find\_friend: поиск друзей

-select\_friend: выбор друга  
- get\_contacts: получение списка контактов с сервера.

- generate\_page: динамически создает кнопки с контактами в зависимости от их количества  
- forward и back: навигация по списку контактов

- open\_saved: открывает сообщения с самим собой

11. Метод сообщений messages\_click  
Главное: содержит два поля ввода, одно для отображения переписки, другое для ввода сообщений. Отправка производится по нажатию на enter.

- send\_message: отправка сообщения на сервер  
- clear\_everything: удаление ненужного переноса строки, который появляется после нажатия на enter, чтобы отправить сообщение

- load\_messages: загружает историю сообщений с сервера. Выполняется в отдельном потоке, чтобы не тормозить работу приложения.  
- updating: обновляет историю сообщений. Если обновлений нет, скроллбары остаются на той позиции, до которой их прокрутил пользователь. Если есть, то история прокручивается до конца вниз.  
- clear\_msghistory: очищает историю сообщений у обоих пользователей  
- go\_back: возвращает в главное меню

12. Метод размещения виджетов главной страницы main\_menu\_place

Сервер

1. Функция получения данных от клиента receive

Аналогична методу receive в клиенте. Принимает аргумент client (это адрес, с которого нужно получить данные). Это позволяет работать с несколькими клиентами.

1. Функция передачи данных клиенту transmit

Аналогична методу transmit в клиенте. Принимает аргументы client и message. Отправляет нужному клиенту нужное сообщение

1. Функция обработки каждого подключения сервера к клиенту handle

После подключения создает переменные connection\_lost (нужна для того, чтобы избежать генерации ошибки) и ID (нужна для для работы с базой данных). Будет работать до тех пор, пока connecton\_lost не примет значение False.

Содержит функцию communication, которая обрабатывает полученные запросы.

* Запрос начинается с «login»: это запрос на вход в аккаунт. Производится поиск в базе данных по логину и паролю, и если совпадение найдено, то значение ID устанавливается и клиенту отправляется подтверждение.
* Запрос начинается с «signup»: запрос на регистрацию. Проверяется уникальность полученных юзернейма и логина. Если все проверки пройдены, то новые данные добавляются в базу данных и клиенту отправляется подтверждение.
* Запрос начинается с «username»: клиенту отправляется юзернейм, найденный по его ID
* Запрос начинается с «editusername»: изменение юзернейма в базе данных
* Запрос начинается с «editpassword»: изменение пароля в базе данных
* Запрос начинается с «sendmsg»: отправка сообщений. Из запроса получаем ID пользователя, которому нужно отправить сообщение. Если ID = None, открываем переписку с самим собой. В противном случае ищем текстовый файл, состоящий из ID клиента и ID нужного пользователя и открываем его. Если такого файла нет, создаем новый
* Запрос начинается с «clearmsghistory»: очистка истории сообщений. Поиск нужного файла происходит по такому же прицнипу, как и в sendmsg
* Запрос начинается с «loadmsghistory»: получение актуальной истории сообщений и передача клиенту
* Запрос начинается с «findfriend»: поиск юзернейма в базе данных
* Запрос начинается с «getcontacts»: анализ всех текстовых файлов на предмет содержания в их названии ID пользователя. Если такие есть, ID самого пользователя игнорируется и сохраняется только ID собеседников. По этим ID производится поиск соответствующих юзернеймов в базе данных и полученные результаты отправляются клиенту.
* Запрос начинается с «getfriendID»: получение ID пользователя по юзернейму
* Запрос начинается с «getpfp»: получение индекса фотографии профиля пользователя из базы данных
* Запрос начинается с «editpfp»: изменение индекса фотографии профиля пользователя в базе данных
* Если запрос не подходит ни под одну из вышеперечисленных команд, выводится сообщение «unknown command».

1. Функция main

Создает TCP-сокет, который будет использоваться для обработки подключений к серверу.  
Хост – локальный ip адрес компьютера  
Запускает бесконечный цикл, который будет принимать клиентов и для каждого клиента создает новый поток, который будет обрабатывать общение с ним