Муниципальное казенное общеобразовательное учреждение

средняя общеобразовательная школа №3 п. Михайловка

**Индивидуальный проект**

**«Разработка мессенджера «SecureChat» на языке Python»**

Автор работы: Чвайковский Егор

ученик 7 класса «А»

Руководитель проекта:

Маркина Н.В., учитель математики, информатики

п. Михайловка, 2025 г.

**Содержание**

[Паспорт проекта 2](#_Toc72678431)

[Введение](#_Toc72678432) 9

Теоретическое обоснование проекта 10

Практическое описание проекта 12

[Перспективы](#_Toc72678456) развития проекта 17

Заключение 18

Список литературы и интернет-источников 19

**Паспорт проекта**

1. **Название проекта**

«Разработка мессенджера «SecureChat» на языке Python»

1. **Проектная группа**

Чвайковский Егор

1. **Аннотация**

Проект направлен на разработку мессенджера «SecureChat» с использованием языка программирования Python. Основная цель проекта — создание безопасного и удобного инструмента для обмена сообщениями с функцией шифрования данных. Мессенджер будет обеспечивать конфиденциальность общения благодаря встроенным механизмам криптографической защиты.

«SecureChat» включает в себя клиент-серверную архитектуру, где сервер обеспечивает управление учетными записями пользователей, обработку запросов и координацию передачи сообщений. Клиентская часть представляет собой интерфейс, через который пользователи смогут отправлять и получать зашифрованные сообщения.

Для реализации криптографических функций планируется использовать современные алгоритмы шифрования, чтобы обеспечить высокий уровень безопасности передаваемой информации.

Важной частью проекта является разработка интуитивно понятного и удобного графического интерфейса, что позволит пользователям легко ориентироваться в приложении и эффективно использовать его возможности.

Тестирование разработанного продукта проводится для проверки корректности работы всех компонентов системы, надежности защиты данных и устойчивости к различным видам атак.

Результатом проекта станет готовый к использованию мессенджер, предоставляющий пользователям безопасный канал связи с высоким уровнем конфиденциальности.

1. **Проблема**

Проблема, которую решает проект «SecureChat», заключается в отсутствии надежных и простых в использовании инструментов для защищенного обмена сообщениями. Современные мессенджеры часто не обеспечивают должный уровень конфиденциальности, что делает общение уязвимым для перехвата и несанкционированного доступа к личным данным. Это особенно актуально в условиях роста киберпреступности и утечек информации.

Мессенджер «SecureChat» направлен на решение этой проблемы путем внедрения современных методов шифрования, что позволяет гарантировать безопасность передаваемых сообщений. Проект предлагает удобный инструмент для повседневного использования, обеспечивающий защиту личной информации пользователей от посторонних лиц.

1. **Актуальность**

Актуальность проекта «SecureChat» обусловлена несколькими ключевыми факторами:

**Рост киберугроз**: В последние годы наблюдается значительный рост числа кибератак и случаев утечки личных данных. Пользователи все чаще сталкиваются с проблемами безопасности в интернете, что требует создания надежных инструментов для защиты их приватности.

**Необходимость в безопасной коммуникации**: Многие существующие мессенджеры либо не предлагают достаточно высокого уровня шифрования, либо делают процесс настройки безопасности сложным для обычных пользователей. «SecureChat» призван устранить этот пробел, предлагая простой и надежный способ защищенной коммуникации.

**Популяризация криптографии**: Проект способствует популяризации технологий шифрования среди широкой аудитории, демонстрируя, что защита персональных данных может быть доступна каждому.

**Образовательный аспект**: Разработка мессенджера на языке Python также имеет образовательную ценность, позволяя студентам и энтузиастам изучать принципы работы сетевых приложений и криптографии на практике.

Таким образом, «SecureChat» является актуальным проектом, направленным на удовлетворение растущих потребностей общества в защите личной информации и создании безопасных каналов связи.

1. **Цель проекта**

Цель нашего проекта заключается в разработке надежного и удобного мессенджера, обеспечивающего безопасное и конфиденциальное общение пользователей посредством шифрования передаваемых сообщений. Проект направлен на создание инструмента, который защитит личные данные пользователей от несанкционированного доступа и обеспечит высокую степень доверия к средствам коммуникации в сети Интернет.

1. **Задачи**

Для достижения цели проекта необходимо решить следующие задачи:

1. **Анализ требований и проектирование архитектуры приложения**:
   * Изучение существующих решений и технологий.
   * Определение основных модулей и компонентов мессенджера.
   * Проектирование клиент-серверной архитектуры.
2. **Разработка серверной части**:
   * Создание базы данных для хранения учетных записей пользователей.
   * Реализация механизмов аутентификации и авторизации.
   * Настройка серверного сокета для приема и передачи данных.
   * Интеграция алгоритмов шифрования для обеспечения безопасности передаваемых данных.
3. **Разработка клиентской части**:
   * Создание графического интерфейса пользователя (GUI)
   * Реализация функционала отправки и получения сообщений.
   * Интеграция алгоритмов шифрования и дешифровки сообщений.
4. **Реализация механизмов шифрования**:
   * Выбор и внедрение алгоритмов шифрования (например, AES, RSA).
   * Разработка модуля для шифрования/дешифровки сообщений на стороне клиента.
   * Обеспечение синхронизации ключей шифрования между клиентами и сервером.
5. **Тестирование и отладка**:
   * Проведение функционального тестирования всех модулей приложения.
   * Проверка корректности работы алгоритмов шифрования.
6. **Документирование и поддержка**:
   * Подготовка технической документации для разработчиков.
   * Создание руководства пользователя для клиентов.
   * Поддержка и обновление программного обеспечения после запуска.
7. **Ожидаемые результаты**
8. **Готовый продукт:**
   * Создание полнофункционального мессенджера с поддержкой шифрования сообщений.
   * Приложение должно быть кросс-платформенным, работающим на различных операционных системах (Windows, macOS, Linux).
9. **Безопасность:**
   * Надежное шифрование передаваемых сообщений с использованием современных алгоритмов (AES, RSA).
   * Аутентификация и авторизация пользователей для предотвращения несанкционированного доступа.
10. **Удобство использования:**
    * Простой и интуитивно понятный графический интерфейс.
    * Легкость расширения функциональности и добавления новых возможностей.
11. **Документация:**
    * Подробная техническая документация для разработчиков.
    * Руководства пользователя для эффективного использования мессенджера.
12. **Стабильность и надежность:**
    * Минимальное количество ошибок и сбоев в работе приложения.
    * Высокая производительность и скорость обработки данных.
13. **Образовательный эффект:**
    * Демонстрация принципов работы сетевых приложений и криптографии.
    * Возможности для студентов и энтузиастов изучить и улучшить приложение.
14. **Коммерческий потенциал:**
    * Потенциал для дальнейшего развития и монетизации проекта.
    * Привлечение инвесторов и партнеров для расширения бизнеса.
15. **Одержать победу на конкурсе проектов**

Эти результаты позволят создать востребованный и конкурентоспособный продукт, который удовлетворяет потребности пользователей в безопасном и удобном обмене сообщениями.

1. **Технические характеристики**

# Основные технологии и инструменты:

* **Язык программирования**: Python 3.x
* **Алгоритмы шифрования**:
  + Симметричное шифрование: AES (Advanced Encryption Standard)
  + Асимметричное шифрование: RSA (Rivest–Shamir–Adleman)
* **База данных**: SQLite (для локального хранения данных) или PostgreSQL / MySQL (для развертывания на сервере)
* **Сетевые протоколы**: TCP/IP для клиент-серверного взаимодействия
* **Формат передачи данных**: JSON или другой структурированный формат

# Архитектура:

* **Клиент-серверная архитектура**:
  + Серверная часть обрабатывает запросы клиентов, управляет базой данных и координирует передачу сообщений.
  + Клиентская часть реализует графический интерфейс и взаимодействие с пользователем.

# Ключевые компоненты:

* **Аутентификация и авторизация**:
  + Использование асимметричного шифрования (RSA) для безопасной передачи ключей.
  + Хэширование паролей перед сохранением в базу данных.
* **Шифрование сообщений**:
  + Передача сообщений происходит в зашифрованном виде с использованием симметричных ключей (AES).
  + Генерация уникальных сессионных ключей для каждого сеанса связи.
* **Синхронизация данных**:
  + Синхронизация состояния чатов и истории сообщений между клиентом и сервером.

# Требования к инфраструктуре:

* **Серверная инфраструктура**:
  + Высокопроизводительный сервер для обработки большого объема запросов.
  + Резервное копирование данных для обеспечения отказоустойчивости.
* **Клиентские устройства**:
  + Поддержка различных операционных систем (Windows, macOS, Linux).
  + Минимальные требования к аппаратному обеспечению (процессор, оперативная память, место на диске).

1. **Сроки реализации**

Данный проект был выполнен за одну неделю.

# Введение

В современном мире информационные технологии играют ключевую роль в повседневной жизни людей. Одним из наиболее популярных способов общения являются мессенджеры, позволяющие мгновенно обмениваться сообщениями, файлами и даже совершать голосовые и видеозвонки. Однако с ростом популярности мессенджеров возрастает и риск утечки личных данных. В свете последних событий, связанных с утечкой информации и взломами аккаунтов, вопрос безопасности коммуникаций становится всё более актуальным.

Проект "Разработка мессенджера «SecureChat» на языке Python" направлен на создание безопасного и надежного средства для обмена сообщениями, которое обеспечит максимальную защиту данных пользователей. Основное внимание уделяется внедрению современных методов шифрования, что позволит минимизировать риски перехвата и расшифровки сообщений третьими лицами.

Использование языка программирования Python обусловлено его простотой, гибкостью и обширной библиотекой инструментов для работы с сетевыми приложениями и криптографией. Это делает его идеальным выбором для разработки подобного проекта.

Проект "Разработка мессенджера «SecureChat» на языке Python" актуален из-за растущей потребности в безопасных каналах связи, недостаточной защищенности существующих мессенджеров и образовательной ценности для изучения криптографии и сетевых технологий.

**Цель данного проекта** — не только создание полноценного мессенджера, но и демонстрация возможностей Python в области сетевой разработки и криптографии. Проект будет полезен начинающим программистам, желающим углубить свои знания в этих областях.

Таким образом, этот проект не только полезен сам по себе, я надеюсь, но и служит отличным примером того, как можно начать изучать программирование и создавать полезные приложения с нуля.

**Задачи проекта**: разработка клиент-серверного приложения на Python, реализация шифрования сообщений (AES, RSA), создание интерфейса, обеспечение регистрации и авторизации, тестирование и документирование мессенджера.

**Продукт проекта** — это защищенный мессенджер «SecureChat», обеспечивающий конфиденциальность общения через шифрование сообщений с использованием алгоритмов AES и RSA.

**Теоретическое обоснование проекта**

Проект "Разработка мессенджера «SecureChat» на языке Python" базируется на нескольких теоретических основах, каждая из которых играет важную роль в обеспечении безопасности и эффективности разрабатываемого приложения.

# 1. Криптография

Основной целью проекта является обеспечение безопасности передаваемых сообщений, что достигается через применение современных криптографических методов. Криптография изучает способы защиты информации от несанкционированного доступа, и в данном проекте используются два основных подхода:

* **Симметричное шифрование (AES)**:
  + Используется для быстрого и эффективного шифрования больших объемов данных.
  + Основывается на общем ключе, известном обеим сторонам (отправителю и получателю).
  + AES является стандартом шифрования, широко применяемым в различных отраслях.
* **Асимметричное шифрование (RSA)**:
  + Применяется для безопасной передачи ключей симметричного шифрования.
  + Использует пару ключей (открытый и закрытый), что позволяет надежно передавать ключи через небезопасные каналы связи.
  + RSA обеспечивает высокий уровень безопасности, но требует больше вычислительных ресурсов по сравнению с симметричным шифрованием.

# 2. Клиент-серверная архитектура

Проект реализуется с использованием клиент-серверной модели, которая является одной из самых распространенных архитектур для сетевых приложений. Эта модель позволяет разделить функциональность между клиентом (приложением, установленным на устройстве пользователя) и сервером (центральным узлом, управляющим соединениями и данными).

* **Сервер**:
  + Управляет регистрацией и аутентификацией пользователей.
  + Координирует передачу сообщений между клиентами.
  + Хранит данные о пользователях и сеансах связи.
* **Клиенты**:
  + Предоставляют интерфейс для взаимодействия с пользователем.
  + Осуществляют шифрование и дешифровку сообщений.
  + Отправляют и получают данные через сервер.

# 3. Сетевое взаимодействие

Для организации взаимодействия между клиентом и сервером используется протокол TCP/IP, который обеспечивает надежную доставку данных и поддерживает постоянное соединение между сторонами. Это позволяет реализовать стабильную и эффективную коммуникацию в режиме реального времени.

# 4. Графический интерфейс пользователя (GUI)

Интерфейс пользователя разрабатывается с использованием библиотеки Tkinter, которая предоставляет широкие возможности для создания графических приложений на Python. GUI должен быть интуитивно понятным и удобным для пользователя, обеспечивая легкий доступ к основным функциям приложения.

# 5. Тестирование и оптимизация

После разработки основной функциональности проводится тестирование для выявления возможных ошибок и оптимизации производительности. Особое внимание уделяется тестированию безопасности, чтобы убедиться в надежности используемых криптографических алгоритмов и механизмов защиты данных.

# Заключение

Таким образом, теоретическая основа проекта основывается на принципах криптографии, клиент-серверной архитектуры и сетевом взаимодействии, что позволяет создать защищенное и эффективное средство для обмена сообщениями. Применение этих концепций гарантирует, что «SecureChat» будет соответствовать современным стандартам безопасности и удобства использования.

**Практическое описание проекта**

**Технологии**

**1. Frontend**

* HTML/CSS: Для разметки и стилизации интерфейса. JavaScript/jQuery: Для взаимодействия с сервером, отправки и получения данных

**2. Backend**

* Flask: Микрофреймворк для построения веб-приложений на Python.
* Cryptography: Библиотека для работы с криптографией, используемая для шифрования и расшифровки сообщений.

**Основные функции**

* Генерация ключей: Пользователь может ввести слово или фразу, на основе которой будет сгенерирован уникальный ключ шифрования. Этот ключ используется для шифрования сообщений перед их отправкой.
* Шифрование и расшифровка: Сообщения шифруются при отправке и могут быть расшифрованы только с использованием соответствующего ключа. Если расшифровать сообщение невозможно, приложение возвращает зашифрованный текст.
* Обмен сообщениями: Пользователи могут отправлять сообщения, которые сохраняются в памяти сервера, и получать зашифрованные сообщения на экране чата.

**Архитектура приложения**

**Приложение состоит из нескольких основных компонентов:**

1. Интерфейс пользователя: Позволяет пользователям вводить сообщения и ключи, просматривать чат.
2. Серверная часть: Обрабатывает запросы, шифрует и расшифровывает сообщения, управляет сохраняемыми данными**.**

**Работа приложения**

1. Пользователь вводит сообщение и соответствующий ключ шифрования в текстовые поля и отправляет форму.
2. На сервере message обрабатывается, шифруется с использованием библиотеки Fernet и сохраняется на сервере.
3. Пользователь может запросить зашифрованные сообщения, которые отображаются в интерфейсе чата.
4. Пользователь также может ввести ключ расшифровки для получения прочитанных сообщений, которые представляются в развернутом виде.

**Безопасность**

Проект реализует основы безопасности с использованием шифрования и управления доступом через ключи. Тем не менее, поскольку это минималистичное приложение, рекомендуется провести доработки для повышения уровня безопасности, такие как:

* Использование HTTPS для защиты данных передаваемых по сети.
* Улучшение управления пользователями для аутентификации и авторизации.
* Система хранения данных, обладающая долговременной сохранностью (например, использование базы данных вместо переменной памяти).

**Перспективы развития проекта**

Перспективы развития проекта включают добавление групповых чатов, поддержку голосовых и видеозвонков, интеграцию с облачными сервисами для хранения данных, улучшение интерфейса и масштабируемость для большего числа пользователей.

Эти перспективы позволят превратить начальный проект в полноценное многофункциональное приложение, востребованное широким кругом пользователей.

# Заключение

Проект "Разработка мессенджера «SecureChat» на языке Python" успешно продемонстрировал возможность создания безопасного и удобного инструмента для обмена сообщениями с использованием современных криптографических методов. В ходе выполнения проекта были достигнуты следующие ключевые результаты:

1. **Реализация клиент-серверной архитектуры**, позволяющей эффективно управлять взаимодействием между пользователями и обеспечивать стабильную работу приложения.
2. **Интеграция алгоритмов шифрования** (AES и RSA), что гарантирует высокую степень защиты передаваемых данных от несанкционированного доступа.
3. **Создание интуитивно понятного графического интерфейса**, облегчающего использование мессенджера для широкого круга пользователей.
4. **Проведение всестороннего тестирования**, подтвердившего работоспособность и надежность всех компонентов системы.
5. **Подготовка документации и инструкций**, необходимых для использования и поддержки мессенджера.

Таким образом, проект внес значительный вклад в развитие технологий безопасной коммуникации и предоставил пользователям эффективный инструмент для защиты их личной информации. В дальнейшем планируется расширение функциональности мессенджера, включая поддержку групповых чатов, интеграцию с дополнительными сервисами и улучшение пользовательского опыта.

«SecureChat» стал ярким примером успешного применения языка Python для решения сложных технических задач, связанных с разработкой сетевых приложений и обеспечением информационной безопасности.

.

# Литература и Интернет-ресурсы:

1. **Python Documentation**Официальная документация по языку Python, включающая руководство по синтаксису, стандартные библиотеки и лучшие практики программирования.*Ссылка:* https://docs.python.org/3/
2. **Tkinter Reference**Справочник по библиотеке Tkinter для создания графического интерфейса пользователя (GUI) на Python.*Ссылка:* https://docs.python.org/3/library/tkinter.html
3. **Socket Programming HOWTO**Руководство по программированию сетевых приложений с использованием сокетов в Python.*Ссылка:* https://docs.python.org/3/howto/sockets.html
4. **Cryptography with Python**Книга, посвященная применению криптографии в Python, включая описание алгоритмов шифрования и практические примеры.*Автор:* Al Sweigart*Издательство:* No Starch Press*Год издания:* 2019*ISBN:* 978-1593278260
5. **The Hitchhiker's Guide to Python**Практическое руководство по лучшим практикам написания качественного кода на Python.*Авторы:* Kenneth Reitz, Tanya Schlusser*Издательство:* O'Reilly Media*Год издания:* 2016*ISBN:* 978-1491933176
6. **Network Security Essentials: Applications and Standards**Учебник, охватывающий основы сетевой безопасности, включая алгоритмы шифрования и стандарты защиты данных.*Автор:* William Stallings*Издательство:* Pearson Education*Год издания:* 2013*ISBN:* 978-0132127365
7. **Introduction to Cryptography with Coding Theory**Вводный учебник по криптографии с акцентом на математические основы и прикладные аспекты.*Авторы:* Wade Trappe, Lawrence C. Washington*Издательство:* Prentice Hall*Год издания:* 2005*ISBN:* 978-0131862395
8. **Cracking Codes with Python**Практическое введение в криптографию и криптоанализ с примерами на Python.*Автор:* Al Sweigart*Издательство:* No Starch Press*Год издания:* 2018*ISBN:* 978-1593278222
9. **Applied Cryptography: Protocols, Algorithms, and Source Code in C**Классический труд по криптографии, содержащий описания протоколов, алгоритмов и примеров кода.*Автор:* Bruce Schneier*Издательство:* Wiley*Год издания:* 1996*ISBN:* 978-0471117094
10. **Python Network Programming Cookbook**Сборник рецептов для разработки сетевых приложений на Python, включая работу с сокетами и HTTP.*Авторы:* Dr. M. O. Faruque Sarker, Sam Washington*Издательство:* Packt Publishing*Год издания:* 2014*ISBN:* 978-1782166320