|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| МИНОБРНАУКИ РОССИИ  ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  «НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ  им. Р.Е. АЛЕКСЕЕВА   |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **Курсовая работа по дисциплине**  **Программирование на языках высокого уровня**   |  | | --- | |  |   *(фамилия, имя, отчество)*   |  |  |  |  | | --- | --- | --- | --- | | Институт | Институт радиоэлектроники и информационных технологий | | | | Кафедра | Компьютерные технологии в проектировании и производстве | | | | Группа | 24-ИСТ-2 | | | | Дата защиты | | «» 2025 г. |  | | Индекс | | |

МИНОБРНАУКИ РОССИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

им. Р.Е. АЛЕКСЕЕВА»

(НГТУ)

|  |  |  |
| --- | --- | --- |
| Институт | Институт радиоэлектроники и информационных технологий | |
| Направление подготовки | 09.03.02 Информационные системы и технологии | |
| Направленность (профиль) образовательной программы | | Информационно-коммуникационные системы и сети |
| Кафедра | Компьютерные технологии в проектировании и производстве | |

|  |
| --- |
| **КУРСОВАЯ РАБОТА**  **по дисциплине Программирование на языках высокого уровня** |

|  |  |  |  |
| --- | --- | --- | --- |
| Студента | Морозкина Дмитрия Александровича | группы | 24-ИСТ-2 |
| на тему |  | | |

Студент:

Изм.

Лист

№ докум.

Подпись

Дата

Лист

2

КР-НГТУ-ИРИТ-ИСТ-24-ИСТ-2-№24-10900

Провер.

Лупанова Е.А.

Разраб.

Морозкин Д.А.

Слепко И.С.

Шеметов Я.Ю.

Лит.

Листов

ЭСВМ 24-ИСТ-2

Морозкин Д.А.

(подпись) (Фамилия И. О.)

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

(дата)

Руководитель:

Лупанова Е.А.

(подпись) (Фамилия И. О.)

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

(дата)

КР защищена \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата)

С оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

**Содержание**

1. [Разработка и анализ технического задания](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark1) 4

Изм.

Лист

№ докум.

Подпись

Дата

Лист

3

КР-НГТУ-ИРИТ-КТПП-24-ИСТ-2-№24-10900

23-06593

* 1. [Цели и задачи проекта](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark2) 4
  2. [Техническое задание на курсовое проектирование 4](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark3)
     1. [Назначение разработки 4](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark4)
     2. [Область применения 4](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark5)
     3. [Требования к функциональным характеристикам 4](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark6)
     4. [Требования к количественным характеристикам 4](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark7)
     5. Требования к техническим средствам 5
     6. Требования к информационной и программной совместимости 5
     7. Требования к интерфейсу пользователя 5
  3. [Анализ технического задания](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark8) 5

1. [Выбор средств для разработки приложения 5](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark11)
   1. [Выбор среды разработки 5](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark12)
   2. [Выбор графической библиотеки](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark13) 5
   3. Выбор архитектуры приложения 6
2. [Разработка приложения](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark14) 6
   1. [Архитектура клиента](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark15) 6
   2. [Описание классов](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark16) 7
      1. [Класс MainWindow](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark17) 7
      2. [Класс auth\_window 10](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark18)
      3. [Класс reg\_window 1](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark19)1
      4. Класс PrivateChatWindow 12
      5. Класс GroupChatWindow 13
      6. Класс TransitWindow 14
      7. Структура UnreadMessage 15
      8. Класс Server 15
   3. [Программная реализация](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark20) 18
      1. [Структура программы 1](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark21)8
      2. [Соглашения по именованию переменных 1](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark22)8
      3. [Входные и выходные данные](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark23) 19
      4. Особенности реализации 19
3. [Тестирование модели 1](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark24)9
   1. Структура тестов 19
      1. Тесты клиентской части 19
      2. Тесты серверной части 20
   2. Результаты тестирования 20
4. [Руководство пользователя 2](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark25)1

[Заключение 2](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark26)4

[Список литературы 2](file:///C:\Users\Димаста\Downloads\2023%20образец%20ПЗ%20Океан.docx#_bookmark26)5

**1 Разработка и анализ технического задания**

**1.1 Цели и задачи проекта**

Целью данного проекта является создание десктопного приложения для чата, которое позволит пользователям взаимодействовать друг с другом.

Объектом исследования является система обмена сообщениями между пользователями.

Предметом исследования — разработка функционала для обмена текстовыми сообщениями, управления групповыми чатами и обеспечения безопасности данных.

**Основные задачи:**

* Разработать пользовательский интерфейс и логику работы приложения.
* Обеспечить взаимодействие между пользователями.
* Реализовать историю сообщений.
* Разработать систему отслеживания персональных сообщений и сообщений общих чатов.

**1.2 Техническое задание на курсовое проектирование**

**1.2.1 Назначение разработки**

Программа предназначена для обмена текстовыми сообщениями между пользователями в реальном времени. Приложение позволяет общаться как один на один, так и в групповых чатах, поддерживает хранение истории сообщений и уведомления о статусе пользователей.

**1.2.2 Область применения**

Данное приложение может быть использовано для коммуникации между пользователями локальной сети. При дальнейшей доработке возможно использование в корпоративной среде для внутренней коммуникации.

**1.2.3 Требования к функциональным характеристикам**

1. Программа должна обеспечивать регистрацию и авторизацию пользователей.
2. Программа должна отображать список пользователей с индикацией их онлайн/оффлайн статуса.
3. Программа должна обеспечивать обмен личными сообщениями между пользователями.
4. Программа должна поддерживать создание групповых чатов и управление участниками.
5. Программа должна сохранять историю сообщений и предоставлять к ней доступ при открытии чата.
6. Программа должна уведомлять о сбоях в подключении и изменении статуса пользователей.

**1.2.4 Требования к количественным характеристикам**

1. Время доставки сообщения - не более 1 секунды.
2. Время загрузки истории сообщений - не более 3 секунд.
3. Время авторизации пользователя - не более 2 секунд.
4. Объем сохраняемой истории сообщений - не менее 7 дней.

**1.2.5 Требования к техническим средствам**

Изм.

Лист

№ докум.

Подпись

Дата

Лист

4

КР-НГТУ-ИРИТ-КТПП-24-ИСТ-2-№24-10900

23-06593

Требования к техническим средствам определяются установленной операционной системой. Программа не является ресурсоемкой и не предъявляет особых требований к аппаратному обеспечению, кроме наличия сетевого подключения.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

5

КР-НГТУ-ИРИТ-КТПП-24-ИСТ-2-№24-10900

23-06593

**1.2.6 Требования к информационной и программной совместимости**

Программное обеспечение должно функционировать на операционных системах семейства MS Windows 7/10/11. Для компиляции и сборки проекта используется система CMake версии 3.16 или выше, что обеспечивает кроссплатформенную совместимость и упрощает процесс разработки. В проекте подключена библиотека Google Test (gtest) для модульного тестирования отдельных компонентов программы, что обеспечивает надежность и стабильность работы приложения. Требуется Qt Framework версии 6.2 или выше для функционирования пользовательского интерфейса и обеспечения сетевого взаимодействия.

**1.2.7 Требования к интерфейсу пользователя**

Интерфейс пользователя должен быть интуитивно понятным и обеспечивать быстрый доступ к основным функциям. Должен включать список контактов с индикацией статуса, окна чатов с историей сообщений и поле для ввода новых сообщений.

**1.3 Анализ технического задания**

Данная программа состоит из двух компонентов - клиентского приложения и серверной части. Сервер обеспечивает аутентификацию пользователей, маршрутизацию сообщений и хранение данных. Клиент предоставляет интерфейс для взаимодействия с системой.

Хранение данных пользователей и истории сообщений реализовано с использованием SQLite. Этот выбор обусловлен простотой интеграции, отсутствием необходимости установки отдельной СУБД и достаточной производительностью для данного типа приложения.

Взаимодействие между клиентом и сервером осуществляется по протоколу TCP/IP с использованием библиотеки QTcpSocket, что обеспечивает надежную доставку сообщений и относительную простоту реализации.

**2. Выбор средств для разработки приложения**

**2.1. Выбор среды разработки**

Для разработки проекта на языке C++ можно использовать различные интегрированные среды разработки (IDE), такие как Qt Creator, Microsoft Visual Studio, CLion и другие.

**Qt Creator** была выбрана для реализации проекта по следующим причинам:

* Полная интеграция с библиотекой Qt, что упрощает разработку приложений с графическим интерфейсом.
* Поддержка кроссплатформенной разработки (Windows, Linux, macOS).
* Удобные инструменты для отладки, профилирования и рефакторинга кода.
* Встроенная поддержка CMake и QMake для управления сборкой проекта.
* Легкость настройки и использования.

Недостатки:

* Большой вес приложений, созданных с использованием Qt.
* Ограниченная функциональность автозавершения кода в некоторых случаях.

**2.2. Выбор графической библиотеки**

Для реализации графического интерфейса приложения были рассмотрены следующие библиотеки: **Qt**, **SFML**, **SDL**.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

6

КР-НГТУ-ИРИТ-КТПП-24-ИСТ-2-№24-10900

23-06593

**Qt** была выбрана для реализации проекта по следующим причинам:

* Поддержка разработки кроссплатформенных приложений.
* Удобные инструменты для создания графического интерфейса.
* Возможность интеграции с Qt Creator.
* Поддержка сигналов и слотов для обработки событий.

Недостатки:

* Большой вес приложений.
* Платная лицензия для коммерческого использования.

**2.3. Выбор архитектуры приложения**

Для реализации проекта будет использоваться клиент-серверная архитектура.

**Клиентская часть:**

* Отвечает за отображение пользовательского интерфейса.
* Обеспечивает отправку и получение сообщений.
* Реализует обработку событий, таких как нажатие кнопок или ввод текста.

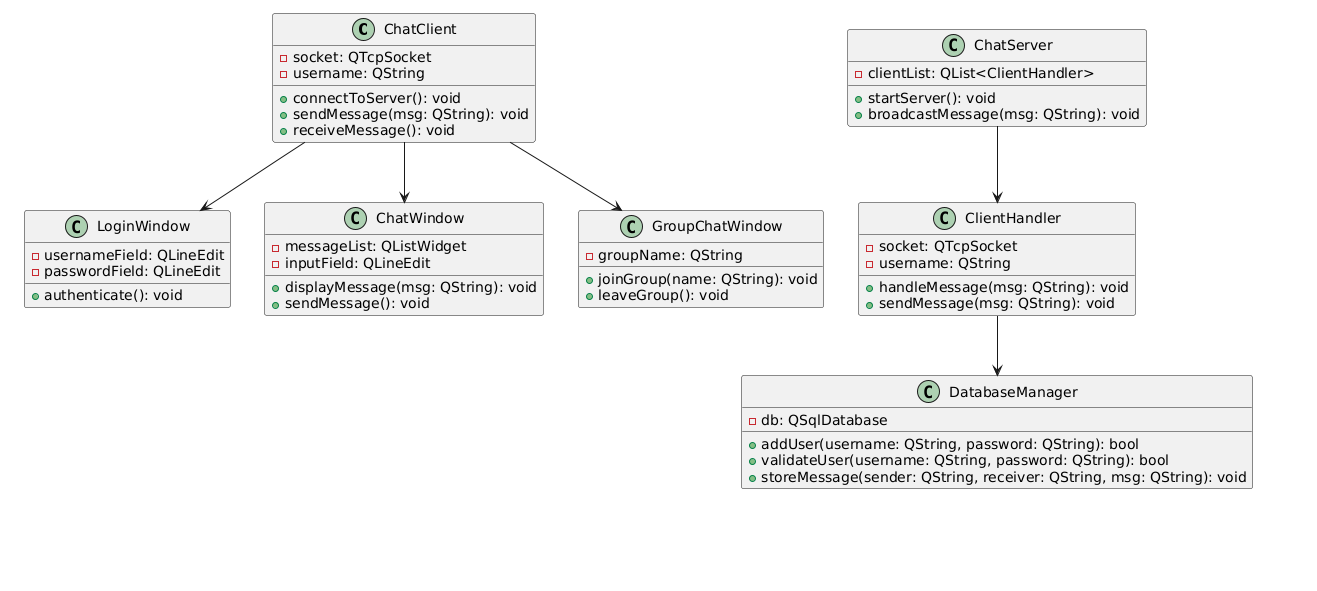
**Серверная часть:**

* Обрабатывает запросы от клиентов.
* Хранит данные пользователей и историю сообщений.
* Обеспечивает маршрутизацию сообщений между клиентами.

**3. Разработка приложения**

Для разработки приложения используется клиент-серверная архитектура. Сервер отвечает за обработку данных, управление пользователями, хранение истории сообщений и управление групповыми чатами. Клиентская часть предоставляет графический интерфейс для взаимодействия с пользователем.

Архитектура приложения представлена на рисунке 1.

****

**Рисунок 1 – Архитектура приложения**

Серверная часть реализована с использованием библиотеки Qt для работы с сетевыми соединениями и базой данных. Клиентская часть также использует Qt для создания графического интерфейса и обработки событий.

**3.1 Архитектура клиента**

В данном приложении было разработано шесть основных классов, обеспечивающих функциональность клиентской части чат-приложения, и одна вспомогательная структура.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

6

КР-НГТУ-ИРИТ-КТПП-24-ИСТ-2-№24-10900

23-06593

Базовый класс программы – класс MainWindow, представляющий главное окно приложения и содержащий основную логику управления соединением с сервером, обработкой сообщений и управлением дочерними окнами.

Классы auth\_window и reg\_window обеспечивают функциональность авторизации и регистрации пользователей соответственно и связаны с классом MainWindow отношениями агрегации.

Классы PrivateChatWindow и GroupChatWindow представляют окна для приватных и групповых чатов соответственно. Оба класса связаны с классом MainWindow отношением ассоциации, поскольку используют его для отправки сообщений на сервер.

Класс TransitWindow представляет диалоговое окно для создания новых групповых чатов и также связан с MainWindow отношением ассоциации.

Структура UnreadMessage, определённая внутри класса MainWindow, используется для хранения непрочитанных сообщений и связана с классом MainWindow отношением композиции.

**3.2 Описание классов**

**3.2.1 Класс MainWindow**

Этот класс является центральным компонентом приложения. Он отвечает за управление соединением с сервером, обработку сообщений, авторизацию пользователей и управление окнами чатов.

К атрибутам класса MainWindow относятся: указатель на интерфейс главного окна, сокет для связи с сервером, коллекции для хранения окон приватных и групповых чатов, данные для авторизации, состояния операций, буфер для обмена данными и таймер для отслеживания таймаутов аутентификации.

class MainWindow : public QMainWindow

{

private:

    Ui::MainWindow \*ui;

    QTcpSocket \*socket;

    QByteArray Data;

    quint16 nextBlockSize;

    auth\_window ui\_Auth;

    reg\_window ui\_Reg;

    QString m\_username;

Изм.

Лист

№ докум.

Подпись

Дата

Лист

7

КР-НГТУ-ИРИТ-КТПП-24-ИСТ-2-№24-10900

23-06593

    QString m\_userpass;

    bool m\_loginSuccesfull;

    QMap<QString, PrivateChatWindow\*> privateChatWindows;

    QMap<QString, GroupChatWindow\*> groupChatWindows;

    enum OperationType { None, Auth, Register };

    OperationType currentOperation;

    QString pendingGroupChatId;

    QMap<QString, QList<UnreadMessage>> unreadMessages;

    QTimer \*authTimeoutTimer;

    Mode mode;

    std::unique\_ptr<MockDatabase> testDb;

};

К методам класса относятся функции для управления соединением с сервером, авторизации и регистрации пользователей, отправки и обработки сообщений, управления окнами чатов, а также вспомогательные методы для тестирования и обработки ошибок.

class MainWindow : public QMainWindow

{

public:

    enum class Mode { Production, Testing };

    explicit MainWindow(QWidget \*parent = nullptr);

Изм.

Лист

№ докум.

Подпись

Дата

Лист

8

КР-НГТУ-ИРИТ-КТПП-24-ИСТ-2-№24-10900

23-06593

    explicit MainWindow(Mode mode, QWidget \*parent = nullptr);

    ~MainWindow();

    void authorizeUser();

    void registerUser();

    void prepareForRegistration();

    void sendMessageToServer(const QString &message);

    void slotReadyRead();

    void clearSocketBuffer();

    bool connectToServer(const QString& host, quint16 port);

    void handlePrivateMessage(const QString &sender, const QString &message);

    void sendPrivateMessage(const QString &recipient, const QString &message);

    void requestPrivateMessageHistory(const QString &otherUser);

    void createGroupChat(const QString &chatName, const QString &chatId);

    void joinGroupChat(const QString &chatId);

    void startAddUserToGroupMode(const QString &chatId);

    void handleAuthenticationTimeout();

    void handleSocketError(QAbstractSocket::SocketError socketError);

    QString getCurrentUsername() const;

    void updateUserList(const QStringList &users);

    void updatePrivateChatStatuses(const QMap<QString, bool> &userStatusMap);

    void testAuthorizeUser(const QString& username, const QString& password);

Изм.

Лист

№ докум.

Подпись

Дата

Лист

9

КР-НГТУ-ИРИТ-КТПП-24-ИСТ-2-№24-10900

23-06593

    bool testRegisterUser(const QString& username, const QString& password);

};

**3.2.2 Класс auth\_window**

Данный класс представляет окно авторизации пользователя. Он содержит поля для ввода логина и пароля, а также кнопки для входа и перехода к регистрации.

class auth\_window : public QDialog

{

public:

    explicit auth\_window(QWidget \*parent = nullptr);

    ~auth\_window();

    QString getLogin();

    QString getPass();

    void LineClear();

    void setButtonsEnabled(bool enabled);

signals:

    void login\_button\_clicked();

    void register\_button\_clicked();

private slots:

    void on\_lineEdit\_textEdited(const QString &arg1);

    void on\_lineEdit\_2\_textEdited(const QString &arg1);

    void on\_Login\_button\_clicked();

    void on\_Register\_button\_clicked();

private:

    Ui::auth\_window \*ui;

    QString m\_username;

Изм.

Лист

№ докум.

Подпись

Дата

Лист

10

КР-НГТУ-ИРИТ-КТПП-24-ИСТ-2-№24-10900

23-06593

    QString m\_userpass;

};

**3.2.3 Класс reg\_window**

Данный класс отвечает за окно регистрации нового пользователя. Он содержит поля для ввода логина, пароля и подтверждения пароля, а также кнопки для регистрации и возврата к окну авторизации.

class reg\_window : public QDialog

{

public:

    explicit reg\_window(QWidget \*parent = nullptr);

    ~reg\_window();

    QString getName();

    QString getPass();

    QString getConfirmPass();

    bool checkPass();

    void ConfirmClear();

    void setButtonsEnabled(bool enabled);

    void setName(const QString& name);

    void setPass(const QString& pass);

    void setConfirmPass(const QString& pass);

signals:

    void register\_button\_clicked2();

    void returnToAuth();

private slots:

    void on\_Password\_line\_2\_textEdited(const QString &arg1);

Изм.

Лист

№ докум.

Подпись

Дата

Лист

11

КР-НГТУ-ИРИТ-КТПП-24-ИСТ-2-№24-10900

23-06593

    void on\_Confirm\_line\_2\_textEdited(const QString &arg1);

Изм.

Лист

№ докум.

Подпись

Дата

Лист

12

КР-НГТУ-ИРИТ-КТПП-24-ИСТ-2-№24-10900

23-06593

    void on\_Register\_button\_2\_clicked();

    void on\_Login\_line\_2\_textEdited(const QString &arg1);

    void on\_return\_auth\_clicked();

private:

    Ui::reg\_window \*ui;

    QString m\_userName;

    QString m\_userPass;

    QString m\_confirmation;

protected:

    void closeEvent(QCloseEvent \*event) override;

};

**3.2.4 Класс PrivateChatWindow**

Данный класс представляет окно для приватного чата между двумя пользователями. Он содержит поле для ввода сообщений, область для отображения истории переписки и кнопку отправки.

class PrivateChatWindow : public QWidget

{

public:

    explicit PrivateChatWindow(const QString &username, MainWindow \*mainWindow, QWidget \*parent = nullptr);

    ~PrivateChatWindow();

    void receiveMessage(const QString &sender, const QString &message);

    void receiveMessage(const QString &sender, const QString &message, const QString &timestamp);

    void setOfflineStatus(bool offline);

    void beginHistoryDisplay();

    void addHistoryMessage(const QString &formattedMessage);

    void endHistoryDisplay();

private slots:

    void on\_sendButton\_clicked();

private:

    Ui::PrivateChatWindow \*ui;

    QString username;

    MainWindow \*mainWindow;

    bool isOffline;

    void sendMessage(const QString &message);

    void updateWindowTitle();

    bool historyDisplayed = false;

    bool statusMessagePending = false;

    bool previousOfflineStatus = false;

    void addStatusMessage();

    QString convertUtcToLocalTime(const QString &utcTimestamp);

};

**3.2.5 Класс GroupChatWindow**

Изм.

Лист

№ докум.

Подпись

Дата

Лист

13

КР-НГТУ-ИРИТ-КТПП-24-ИСТ-2-№24-10900

23-06593

Данный класс представляет окно для группового чата. В отличие от PrivateChatWindow, он дополнительно содержит список участников группы и кнопки для управления составом участников.

class GroupChatWindow : public QWidget

{

public:

    explicit GroupChatWindow(const QString &chatId, const QString &chatName, MainWindow \*mainWindow, QWidget \*parent = nullptr);

    ~GroupChatWindow()

    void receiveMessage(const QString &sender, const QString &message);

    void receiveMessage(const QString &sender, const QString &message, const QString &timestamp);

    void beginHistoryDisplay();

    void addHistoryMessage(const QString &formattedMessage);

    void endHistoryDisplay();

    void updateMembersList(const QStringList &members);

    void setCreator(const QString &creator);

private slots:

    void on\_pushButton\_2\_clicked();

    void on\_lineEdit\_returnPressed();

    void on\_pushButton\_clicked();

    void on\_pushButton\_3\_clicked();

private:

    Ui::Form \*ui;

Изм.

Лист

№ докум.

Подпись

Дата

Лист

14

КР-НГТУ-ИРИТ-КТПП-24-ИСТ-2-№24-10900

23-06593

    QString chatId;

    QString chatName;

    MainWindow \*mainWindow;

    QString creator;

    bool isCreator = false;

    void sendMessage(const QString &message);

    void updateWindowTitle();

    bool historyDisplayed = false;

    QString convertUtcToLocalTime(const QString &utcTimestamp);

};

**3.2.6 Класс TransitWindow**

Данный класс представляет диалоговое окно для создания нового группового чата. Он содержит поле для ввода названия чата и кнопки для создания чата или отмены операции.

class TransitWindow : public QDialog

{

public:

    explicit TransitWindow(MainWindow \*mainWindow, QWidget \*parent = nullptr);

    ~TransitWindow();

private slots:

    void on\_createButton\_clicked();

    void on\_cancelButton\_clicked();

private:

    Ui::TransitWindow \*ui;

Изм.

Лист

№ докум.

Подпись

Дата

Лист

15

КР-НГТУ-ИРИТ-КТПП-24-ИСТ-2-№24-10900

23-06593

    MainWindow \*mainWindow;

};

**3.2.7 Структура UnreadMessage**

Данная структура хранит информацию о непрочитанных сообщениях и определена внутри класса MainWindow.

**struct UnreadMessage {**

**QString sender;**

**QString message;**

**QString timestamp;**

**};**

**3.2.8 Класс Server**

Серверная часть представлена единым классом Server, который наследуется от класса QTcpServer библиотеки Qt и обеспечивает всю необходимую функциональность для обработки подключений клиентов, хранения данных в базе данных SQLite и управления сообщениями.

class Server : public QTcpServer

{

public:

    Server();

    virtual ~Server();

    bool connectDB();

    bool initializeDatabase();

    bool initUserTable();

    bool initMessageTable();

    bool initHistoryTable();

    bool initGroupChatTables();

Изм.

Лист

№ докум.

Подпись

Дата

Лист

16

КР-НГТУ-ИРИТ-КТПП-24-ИСТ-2-№24-10900

23-06593

    QSqlDatabase& getDatabase();

    bool authenticateUser(const QString &username, const QString &password);

    bool registerUser(const QString &username, const QString &password);

    void SendToCllient(QString str);

    bool sendPrivateMessage(const QString &recipientUsername, const QString &message);

    void storeOfflineMessage(const QString &sender, const QString &recipient, const QString &message);

    void sendStoredOfflineMessages(const QString &username, QTcpSocket\* socket);

    void sendUserList(QTcpSocket\* clientSocket);

    void broadcastUserList();

    QString getUsernameBySocket(QTcpSocket \*socket);

    bool createGroupChat(const QString &chatId, const QString &chatName, const QString &creator);

    bool addUserToGroupChat(const QString &chatId, const QString &username);

    bool removeUserFromGroupChat(const QString &chatId, const QString &username);

    void sendGroupChatInfo(const QString &chatId, QTcpSocket \*socket);

    void sendUserGroupChats(const QString &username, QTcpSocket \*socket);

    bool saveGroupChatMessage(const QString &chatId, const QString &sender, const QString &message);

    void sendGroupChatMessage(const QString &chatId, const QString &sender, const QString &message);

    bool logMessage(const QString &sender, const QString &recipient, const QString &message);

Изм.

Лист

№ докум.

Подпись

Дата

Лист

17

КР-НГТУ-ИРИТ-КТПП-24-ИСТ-2-№24-10900

23-06593

    bool saveToHistory(const QString &sender, const QString &message);

    void sendMessageHistory(QTcpSocket\* clientSocket);

    void sendPrivateMessageHistory(QTcpSocket\* clientSocket, const QString &user1, const QString &user2);

    void sendGroupChatHistory(const QString &chatId, QTcpSocket \*socket);

    bool testRegisterUser(const QString &username, const QString &password);

    bool testAuthenticateUser(const QString &username, const QString &password);

    void testSendToClient(const QString &str);

    bool testLogMessage(const QString &sender, const QString &recipient, const QString &message);

    bool testSaveToHistory(const QString &sender, const QString &message);

protected:

    void incomingConnection(qintptr socketDescriptor) override;

private slots:

    void slotReadyRead();

    void clientDisconnected();

private:

    QTcpSocket \*socket;

    QByteArray Data;

    quint16 nextBlockSize;

    QVector<QTcpSocket\*> Sockets;

    QSqlDatabase srv\_db;

    struct AuthenticatedUser {

        QString username;

        QTcpSocket\* socket;

Изм.

Лист

№ докум.

Подпись

Дата

Лист

18

КР-НГТУ-ИРИТ-КТПП-24-ИСТ-2-№24-10900

23-06593

        bool isOnline;

    };

    QList<AuthenticatedUser> authenticatedUsers;

};

**3.3 Программная реализация**

**3.3.1 Структура программы**

Программа клиентской части чат-приложения реализована в виде нескольких модулей, каждый из которых отвечает за определенную функциональность.

Программа состоит из следующих основных модулей:

1. **mainwindow** – главное окно приложения, управление соединением и обработка сообщений
2. **auth\_window** – окно авторизации пользователя
3. **reg\_window** – окно регистрации нового пользователя
4. **privatechatwindow** – окно приватного чата
5. **groupchatwindow** – окно группового чата
6. **transitwindow** – диалоговое окно создания группового чата
7. **server** – серверная логика

Каждый модуль состоит из заголовочного файла (.h) с объявлением класса и файла реализации (.cpp) с определением методов.

Проект также включает файлы UI-форм (.ui), разработанные в Qt Designer, для каждого окна приложения.

**3.3.2 Соглашения по именованию переменных**

В ходе разработки клиентского приложения были выработаны следующие соглашения по именованию и написанию кода:

1. Имена классов начинаются с заглавной буквы (например, MainWindow, PrivateChatWindow).
2. Имена методов классов также начинаются с заглавной буквы (например, SendToServer, GetCurrentUsername).
3. Приватные атрибуты классов имеют префикс m\_ (например, m\_username, m\_userpass).
4. Локальные переменные имеют осмысленные названия, начинающиеся со строчной буквы.
5. В коде используются комментарии для пояснения сложных алгоритмов и блоков кода.
6. Текст программы структурирован, блоки кода отделены друг от друга пустыми строками.
7. Вложенные операторы имеют дополнительный отступ для улучшения читаемости кода.

**3.3.3 Входные и выходные данные**

Изм.

Лист

№ докум.

Подпись

Дата

Лист

19

КР-НГТУ-ИРИТ-КТПП-24-ИСТ-2-№24-10900

23-06593

Входными данными для приложения являются:

1. Учетные данные пользователя (логин и пароль) для авторизации или регистрации
2. Текстовые сообщения, вводимые пользователем для отправки собеседникам
3. Названия создаваемых групповых чатов
4. Данные, получаемые от сервера (списки пользователей, истории сообщений и т.д.)

Выходными данными являются:

1. Отображаемые в окнах чатов сообщения
2. Списки пользователей и групповых чатов в главном окне
3. Статусы пользователей (онлайн/оффлайн)
4. Информационные сообщения и диалоги для пользователя

**3.3.4 Особенности реализации**

Для защиты от проблем с соединением реализован механизм таймаутов при авторизации и регистрации, а также обработка различных сетевых ошибок.

Приложение поддерживает два режима работы: обычный (Production) и тестовый (Testing), что позволяет проводить автоматизированное тестирование функциональности.

**4 Тестирование модели**

Для тестирования устойчивости и корректности работы ChatApp применяется комплексная система автоматических тестов. Тесты интегрированы в CI-конвейер с помощью GitHub Actions, что позволяет контролировать качество кода при каждом внесении изменений.

**4.1 Структура тестов**

В проекте реализованы два основных набора тестов:

* **Тесты клиентской части** - проверяют работу пользовательского интерфейса и бизнес-логики клиента
* **Тесты серверной части** - проверяют функциональность сервера, обработку запросов и управление данными

**4.1.1 Тесты клиентской части**

Клиентские тесты (tests/test\_client) включают следующие группы:

Изм.

Лист

№ докум.

Подпись

Дата

Лист

20

КР-НГТУ-ИРИТ-КТПП-24-ИСТ-2-№24-10900

23-06593

1. **Тесты основного окна** (MainWindowTest)
   * TestInitialState - проверяет начальное состояние главного окна
   * TestAuthentication - тестирует процесс аутентификации пользователя
   * TestRegistration - проверяет функцию регистрации новых пользователей
   * TestUserListUpdate - тестирует обновление списка пользователей
   * TestPrivateChatCreation - проверяет создание и работу приватных чатов
2. **Тесты интерфейса аутентификации** (AuthUITests)
   * AuthWindowInputHandling - тестирует ввод данных в окне авторизации
   * RegWindowPasswordConfirmation - проверяет подтверждение пароля в окне регистрации

**4.1.2 Тесты серверной части**

Серверные тесты (tests/test\_server) включают:

1. **Основные тесты сервера** (ServerTest)
   * ConnectDB - проверяет подключение к базе данных
   * RegisterUser - тестирует регистрацию пользователей
   * AuthenticateUser - проверяет аутентификацию
   * SendToClient - тестирует отправку сообщений клиентам
   * LogMessage - проверяет логирование сообщений
   * SaveToHistory - тестирует сохранение истории сообщений
2. **Тесты аутентификации** (ServerAuthenticationTests)
   * AuthenticatesValidUser - проверяет успешную аутентификацию
   * RejectsInvalidCredentials - тестирует отказ при неверных учетных данных
3. **Тесты регистрации** (ServerRegistrationTests)
   * RegistersNewUser - проверяет успешную регистрацию новых пользователей
   * RejectsDuplicateUsername - тестирует отказ при дублировании имени пользователя

**4.2 Результаты тестирования**

Результаты выполнения тестов доступны в GitHub Actions по ссылке: [https://github.com/Olovat/ChatApp/actions](vscode-file://vscode-app/c:/Users/User/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html)

На странице Actions отображается:

* Статус выполнения каждого теста
* Журнал выполнения сборки и тестирования

Система непрерывной интеграции запускает полный набор тестов при каждом изменении кода.

**5. Руководство пользователя**

Чтобы начать пользоваться нашим чатом, сначала нужно скачать программу на свой компьютер. Для этого перейдите по ссылке <https://github.com/Olovat/ChatApp> и скопируйте репозиторий себе. После скачивания найдите на своем компьютере папку сервера по пути server/build/Desktop\_Qt\_6\_8\_2\_MinGW\_64\_bit-Debug/debug и запустите файл server.exe. У вас откроется окно сервера - его нужно обязательно оставить работать, так как без него чат не сможет функционировать.

Теперь перейдите в папку клиента по адресу client/build/Desktop\_Qt\_6\_8\_2\_MinGW\_64\_bit-Debug/debug и откройте файл client.exe. Перед вами появится окно авторизации. Если вы уже регистрировались ранее, просто введите свой логин и пароль. Если вы новый пользователь, нажмите кнопку "Зарегистрироваться" и заполните необходимые данные для создания аккаунта.

Изм.

Лист

№ докум.

Подпись

Дата

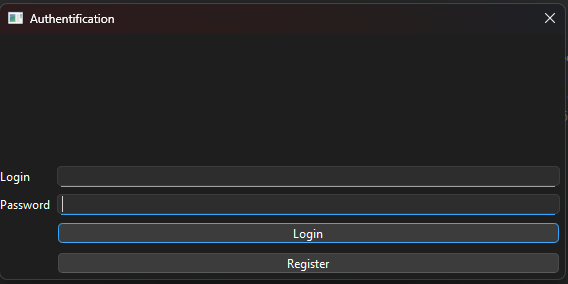
Лист

21

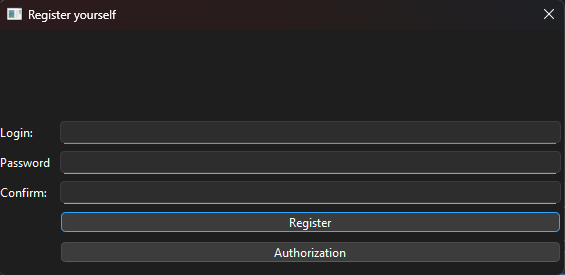
КР-НГТУ-ИРИТ-КТПП-24-ИСТ-2-№24-10900

23-06593

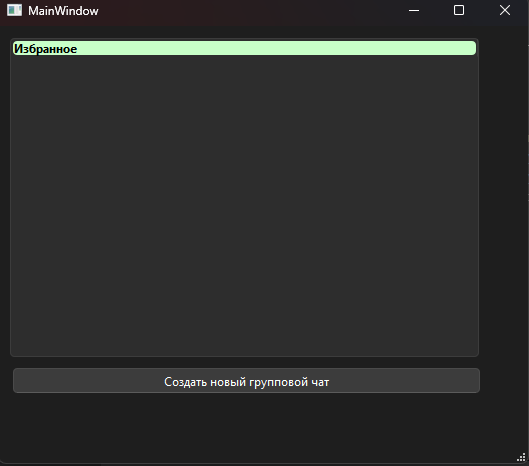
После успешного входа вы попадете в главное меню, где увидите список всех доступных чатов. Здесь можно выбрать любой чат и начать общение. Обратите внимание на цветовые подсказки: пользователи, которые сейчас онлайн, выделены зеленым цветом, те, кто не в сети - серым. Групповые чаты отмечены синим цветом, а новые сообщения и уведомления подсвечиваются желтым. Если вы хотите создать новый групповой чат, просто нажмите соответствующую кнопку и добавьте в него нужных участников из списка контактов.



Окно авторизации



Окно регистрации



Изм.

Лист

№ докум.

Подпись

Дата

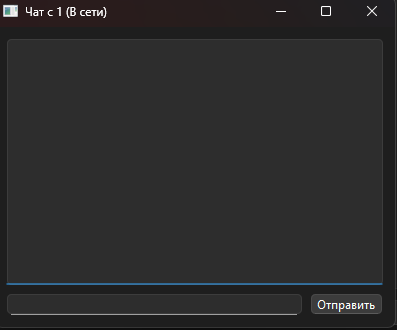
Лист

22

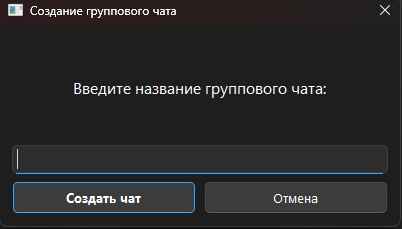
КР-НГТУ-ИРИТ-ГИС-24-ИСТ-2-№24-10900

23-06593

Список пользователей



Окно чата с пользователем



Изм.

Лист

№ докум.

Подпись

Дата

Лист

23

КР-НГТУ-ИРИТ-КТПП-24-ИСТ-2-№24-10900

23-06593

Окно создание группового чата

**Заключение**

В результате выполнения курсовой работы были решены следующие задачи:

1) проведено исследование предметной области;

2) выявлены основные требования, предъявляемые к проекту;

3) проведен сравнительный анализ программных средств для разработки приложения;

4) разработана архитектура приложения;

5) разработана диаграмма классов для приложения;

Изм.

Лист

№ докум.

Подпись

Дата

Лист

24

КР-НГТУ-ИРИТ-КТПП-24-ИСТ-2-№24-10900

23-06593

6) разработан код приложения;

7) выполнена анализ входных параметров приложения на устойчивость модели

8) разработана документация пользователя.

Разработанное приложение имеет простой и понятный интерфейс, который позволяет пользователю увидеть всю информацию о текущем состоянии модели. Есть возможность изменить стартовые настройки модули без перекомпиляции приложения.

Недостатками разработанного приложения являются:

Дальнейшее развитие системы будет направлено на устранение существующих недостатков. Разработанное приложение полностью соответствует требованиям технического задания. Задача, поставленная на курсовое проектирование, полностью выполнена.

**Список литературы:**

Изм.

Лист

№ докум.

Подпись

Дата

Лист

25

КР-НГТУ-ИРИТ-КТПП-24-ИСТ-2-№24-10900

23-06593

1. Справочник по С++. [Электронный ресурс] / Режим доступа: https://ru.cppreference.com/w/ /
2. Язык UML: что это такое и зачем он нужен. [Электронный ресурс] / Режим доступа: https://skillbox.ru/media/code/yazyk-uml-chto-eto-takoe-i-zachem-on- nuzhen/ /
3. клиент-сервер в QT | QTcpServer просто! <https://www.youtube.com/watch?v=QqwlCczvnIc&t=1s>