**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc105594730)

[1 АНАЛИЗ ИСХОДНЫХ ДАННЫХ 4](#_Toc105594731)

[1.1 Анализ предметной области 4](#_Toc105594732)

[1.2 Анализ аналогов и прототипов 4](#_Toc105594732)

[1.3 Специальное и общесистемное программное обеспечение 8](#_Toc105594733)

[2 ПРОГРАММНОЕ ПРОЕКТИРОВАНИЕ 11](#_Toc105594734)

[2.1 Функции клиента и сервера 11](#_Toc105594735)

[2.2 Макет пользовательского интерфейса 11](#_Toc105594736)

[2.3 Прикладной протокол взаимодействия клиента и сервера 13](#_Toc105594737)

[2.4 Алгоритм работы клиента и сервера 14](#_Toc105594738)

[3 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ И ТЕСТИРОВАНИЕ 15](#_Toc105594739)

[3.1 Детальная реализация функциональных частей программного обеспечения 15](#_Toc105594740)

[3.1.1 Реализация функциональных частей клиента и сервера 15](#_Toc105594741)

[3.1.2 Реализация пользовательского интерфейса клиента 18](#_Toc105594742)

[3.2 Сопроводительная документация программного обеспечения 19](#_Toc105594743)

[3.2.1 Описание установки и запуска приложения 19](#_Toc105594744)

[3.2.2 Руководства пользователя 20](#_Toc105594745)

[3.3 Тестирование прогаммного обеспечения 20](#_Toc105594746)

[ЗАКЛЮЧЕНИЕ 23](#_Toc105594747)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 24](#_Toc105594748)

[ПРИЛОЖЕНИЕ А Диаграмма вариантов использования 25](#_Toc105594750)

[ПРИЛОЖЕНИЕ Б Диаграмма взаимодействия 26](#_Toc105594751)

[ПРИЛОЖЕНИЕ В Диаграмма компонентов 27](#_Toc105594752)

# Введение

Современные образовательные учреждения сталкиваются с вызовами, связанными с оценкой знаний студентов. Традиционные методы, такие как письменные тесты и экзамены, требуют значительных ресурсов и времени, а также подвержены субъективности. В условиях цифровизации возникает потребность в инструментах, которые обеспечат объективную, оперативную и удобную проверку знаний.

Клиент-серверное приложение для тестирования, разработанное для работы в локальной сети университета, решает эти проблемы. Оно позволяет проводить тесты в аудитории при защищенной сети, гарантируя, что данные не покинут пределы университета. Это обеспечивает высокую безопасность и конфиденциальность информации, а также независимость от внешних интернет-сервисов.

Проектируемое приложение автоматизирует процесс тестирования, обеспечивает мгновенную проверку результатов и исключает субъективность оценивания. Удобный интерфейс делает его доступным и для студентов, и для преподавателей, минимизируя время на освоение системы и повышая эффективность учебного процесса.

Цель курсового проекта заключается в разработке оконного приложения для проверки и оценки знаний студентов в локальной сети университета.

Для реализации курсового проекта будет использоваться язык программирования C++. Он широко применяется для разработки программного обеспечения, являясь одним из самых популярных и мощных языков программирования.

Кроме того, для создания графического интерфейса приложения будет использоваться Qt Creator. Qt Creator является мощной и удобной средой разработки, поддерживающей C++ и предоставляющей все необходимые инструменты для создания кроссплатформенных оконных приложений с богатым пользовательским интерфейсом.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Изучение и анализ требований образовательных учреждений к безопасности и конфиденциальности данных;
2. Проектирование архитектуры клиент-серверного приложения для работы в локальной сети.;
3. Разработка серверной части приложения для управления тестами, результатами и учетными записями пользователей.;
4. Разработка клиентской части с удобным интерфейсом для студентов и преподавателей.;
5. Создание документации, включая техническое описание, инструкции по установке и использованию, а также руководство по администрированию.

# 1 Анализ исходных данных

## **Анализ предметной области**

Задача проекта – это создание оконного приложения, представляющего собой клиент-серверное приложение “Платформа для проверки знаний”.

Большая часть современных образовательных процессов переходит в онлайн, поэтому разработка надежного клиент-серверного приложения для проверки и оценки знаний имеет решающее значение для обеспечения объективности и безопасности учебного процесса.

Приложение для проверки и оценки знаний – это система, позволяющая проводить тестирование студентов в защищенной сети, гарантируя конфиденциальность и целостность данных. В данном приложении серверная часть управляет готовыми тестами и результатами с помощью файловой системы, а клиентская часть предоставляет интерфейс для студентов и преподавателей для взаимодействия с системой. Тесты создаются заранее и загружаются в систему администратором.

Файловая система на сервере используется для хранения и управления тестами и результатами, что позволяет упростить структуру приложения и облегчить его поддержку. Клиентская часть приложения предоставляет удобный интерфейс для выполнения различных действий, таких как прохождение тестов и отправка результатов на сервер для последующего анализа администратором.

Приложение, работающее в локальной защищенной сети, обеспечивает надежную и безопасную среду для проведения онлайн тестирования, что особенно важно для обеспечения академической честности и предотвращения утечек данных.

В данном курсовом проекте будет использоваться транспортный протокол TCP. Это название происходит от сокращения английских терминов Transmission Control Protocol и Internet Protocol. Этот набор правил позволяет осуществлять сквозную передачу данных, а также управлять различными аспектами этого процесса. В него входят такие функции, как формирование пакетов, их отправка, получение, маршрутизация, распаковка и передача программному обеспечению. Стек протоколов TCP/IP был разработан в 1972 году на основе протокола NCP (Network Control Protocol), и в январе 1983 года он стал официальным стандартом для интернета. [1]

Исходные данные для разработки данного курсового проекта:

1. Задача сервера – управление готовыми тестами и результатами с использованием файловой системы. Сервер обеспечивает безопасное хранение данных и выполнение команд от клиентов;
2. Задача клиента – обеспечение интерфейса для студентов и преподавателей для прохождения тестов и отправки результатов на сервер. Клиент должен быть удобным и интуитивно понятным для пользователей.;
3. Конфигурация сетевого адреса – в специальном файле;
4. Клиентский интерфейс – оконный, разработанный с использованием Qt;
5. Тип сокетов – блокирующий;
6. Транспортный протокол – TCP;
7. Язык программирования – С++;
8. Вспомогательные библиотеки – Qt для создания графического интерфейса;
9. Платформа реализации – ОС Windows;
10. Для работы с файловыми объектами использовать системные вызовы;
11. Выводить информационные сообщения в стандартный поток вывода на сервере;

Основной задачей при проектировании программы является простота использования для пользователя и работоспособность программы.

Необходимо выделить основные задачи проектирования программы:

1. Спроектировать внешнюю и внутреннюю организацию клиентской и серверной части программы.
2. Организовать удобный оконный интерфейс для студентов и преподавателей с использованием библиотеки Qt.
3. Реализовать серверную часть, которая будет управлять тестами и результатами, храня их в файловой системе.
4. Реализовать клиентскую часть, которая будет предоставлять интерфейс для прохождения тестов и отправки результатов.
5. Тестировать программу для проверки работоспособности клиента и сервера, а также для поиска и исправления возможных ошибок.
6. Создать документацию, включающую техническое описание системы, инструкции по установке и использованию, а также руководство по администрированию.

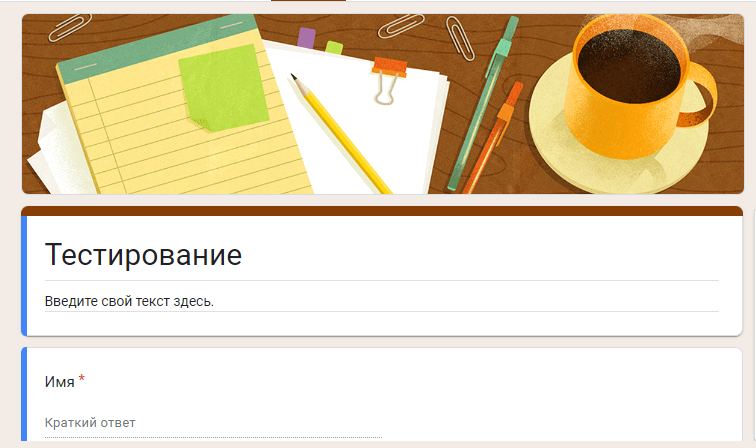
Управление системой будет осуществляться при помощи пользовательского интерфейса, разработанного с использованием библиотеки Qt. Пользователь выбирает, какую операцию требуется выполнить в данный момент из меню приложения. Имеется возможность загружать готовые тесты, просматривать результаты тестирования и управлять настройками системы, в то время как студенты могут проходить тесты, получая удобный и интуитивно понятный интерфейс для взаимодействия с системой.

## **Анализ аналогов и прототипов**

В качестве аналогов проектирования системы рассмотрим следующие решения:

1. Google Forms – позволяет создавать опросы и тесты с различными типами вопросов, такими как выбор ответа, короткий ответ, заполнение пропусков и другие. Результаты могут быть автоматически собраны и анализированы через Google Sheets. [2]

Сервис Google Forms является бесплатным и доступным для всех пользователей с аккаунтом Google. Это делает его популярным выбором для образовательных учреждений. Автоматическая обработка ответов и мощные аналитические инструменты Google Sheets позволяют эффективно обрабатывать и анализировать собранные данные. Google Forms представлен на рисунке 1.1.



**Рисунок 1.1** – Сервис Google Forms

Плюсы:

–Удобство использования.

–Автоматический сбор и анализ данных.

–Доступность и бесплатность.

Минусы

–Ограниченная функциональность

–Проблемы с конфиденциальностью

–Отсутствие адаптивного дизайна

–Зависимость от интернет-соединения

2. Moodle – это платформа для управления обучением (LMS), которая предоставляет возможности создания тестов, опросов и других учебных материалов. Moodle также позволяет администраторам отслеживать активность студентов и оценивать их успеваемость. Кроме того, Moodle позволяет добавлять мультимедийные элементы в тесты, что делает обучение более интерактивным и увлекательным. Платформа предоставляет инструменты для автоматической проверки ответов и выставления оценок, а также для детального анализа результатов учащихся. [3]

Moodle является бесплатным и с открытым исходным кодом, что делает его доступным для всех образовательных учреждений и организаций. Эта платформа широко используется в школах, университетах и корпоративных учебных центрах по всему миру. Moodle поддерживает совместную работу, позволяя преподавателям и студентам взаимодействовать через форумы, чаты и другие средства коммуникации. Администраторы могут легко управлять курсами, контролировать доступ к материалам и генерировать отчеты о прогрессе студентов. Moodle также интегрируется с различными другими системами и инструментами, что делает его гибким и масштабируемым решением для различных образовательных потребностей. На рисунке 1.2 представлен сервис Moodle.



**Рисунок 1.2** – Сервис Moodle

Плюсы:

– Расширенные возможности управления обучением

– Гибкость и настраиваемость

– Мониторинг и анализ

Минусы:

– Сложность настройки

– Необходимость в постоянной поддержке и обновлениях

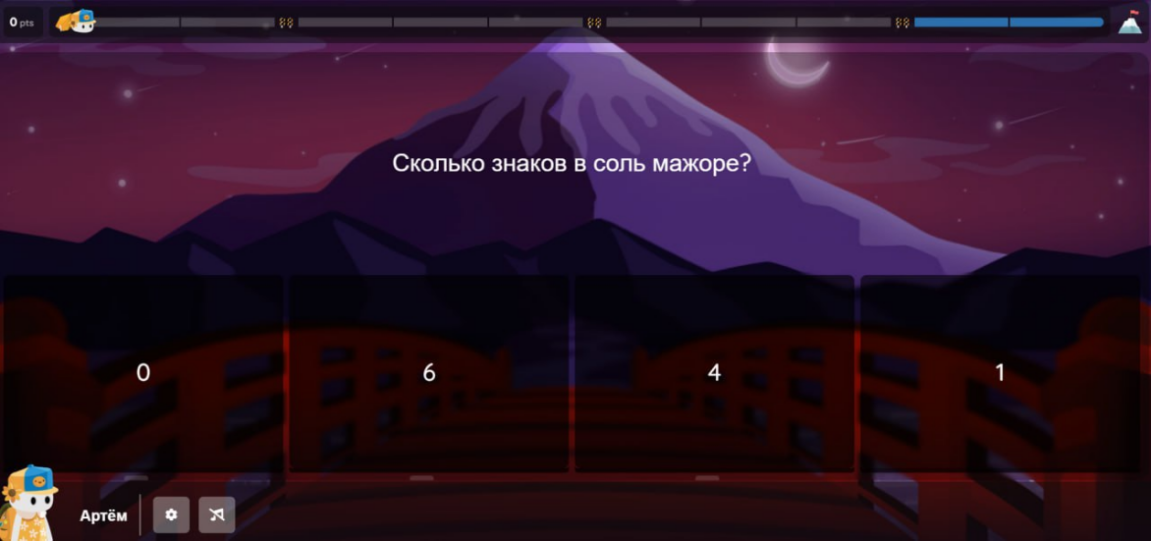
– Сложность использования для преподавателей

– Ограниченные интеграционные возможности

– Необходимость в высокоскоростном интернет-соединении

3. Quizizz – это интерактивная платформа для создания и проведения онлайн-тестов и игр. Она предлагает широкий спектр возможностей для персонализации тестов и мгновенного получения обратной связи. [4]

Quizizz предоставляет возможность создавать тесты и игры с различными вариантами вопросов и интерактивными форматами. Благодаря широкому набору инструментов для персонализации, пользователи могут адаптировать контент под свои образовательные потребности, делая процесс обучения более увлекательным и интересным. Эта платформа также обеспечивает мгновенную обратную связь, что помогает студентам быстро оценить свой прогресс и идентифицировать слабые места для дальнейшего улучшения. Сервис Quizizz представлен на рисунке 1.3.



**Рисунок 1.3** – Сервис Quizizz

## **Специальное и общесистемное программное обеспечение**

Для разработки клиент-серверного приложения " Платформа для проверки знаний" был выбран язык программирования C++. C++ – это мощный и универсальный язык, который объединяет возможности языка C с дополнительными функциями объектно-ориентированного программирования. Благодаря статической типизации C++, ошибки могут быть обнаружены на этапе компиляции, что способствует повышению производительности приложения.

Язык C++ поддерживает множество ключевых особенностей, таких как классы и объекты, наследование, полиморфизм, абстрактные классы, шаблоны, перегрузка операторов и функций, обработка исключений и стандартная библиотека шаблонов (STL). Эти возможности позволяют разработчикам создавать модульные, гибкие и эффективные программы. C++ широко используется в системном программировании, разработке игр, встраиваемых системах и создании высокопроизводительных приложений. Он позволяет тесно взаимодействовать с аппаратным обеспечением и обеспечивает высокую производительность и эффективное использование ресурсов. [2]

Одним из значительных преимуществ C++ является его совместимость с языком C. Это означает, что код, написанный на C, может быть интегрирован с кодом на C++, что обеспечивает плавный переход и расширение функциональности существующих программ. C++ также предоставляет высокую степень контроля над системными ресурсами, что делает его идеальным выбором для разработки приложений, требующих высокой производительности и эффективности. [2]

На основании этих характеристик можно сделать вывод, что язык C++ является удобным и подходящим инструментом для разработки клиент-серверного приложения. Он предоставляет все необходимые средства для создания сложных и высокопроизводительных систем, а также позволяет разработчикам эффективно использовать свои навыки в программировании на C++.

Для разработки клиент-серверного приложения «Специализированный календарь» были предложены следующие технологии разработки графического интерфейса пользователя (далее GUI):

– WinForms;

– QT.

Qt Creator является одной из наиболее популярных кроссплатформенных интегрированных сред разработки (IDE). Основные ее преимущества включают удобство использования, высокую скорость работы и доступность, так как это программное обеспечение с открытым исходным кодом. В Qt Creator поддерживаются языки программирования C, C++ и QML. [5]

Основная цель данной IDE – обеспечить быструю и эффективную кроссплатформенную разработку, используя собственный фреймворк Qt. Это позволяет разработчикам создавать приложения, не разрабатывая их отдельно для каждой операционной системы, а писать общий код и адаптировать его под специфические особенности различных платформ.

Qt Creator также включает инструмент Qt Designer, который позволяет проектировать пользовательский интерфейс приложений, добавляя и перемещая элементы визуально. В качестве систем сборки в Qt Creator используются qmake, CMake и autotools, что обеспечивает гибкость и мощные возможности для управления проектами. Благодаря этим функциям Qt Creator становится идеальной средой для разработки сложных приложений с привлекательным и функциональным интерфейсом.

WinForms (Windows Forms) – это технология, используемая для создания графических интерфейсов пользователей (GUI) в среде разработки Visual Studio 2022, ориентированная на разработку приложений для операционной системы Windows. С помощью WinForms разработчики могут легко создавать интерактивные приложения, используя разнообразные компоненты и элементы управления, такие как кнопки, текстовые поля, таблицы, меню и многое другое. [6]  
 Сравнение данных GUI представлено в таблице 1.1.

**Таблица 1.1 – Возможности GUI**

|  |  |  |
| --- | --- | --- |
| Категория | QT | WinForms |
| Обучаемость | + Хорошая документация и ресурсы для обучения, поддержка сообщества | - Легко изучаемая, особенно для начинающих разработчиков |
| Документация и поддержка | + Широкая документация и активное сообщество | + Обширная документация и поддержка от Microsoft |
| Визуальные эффекты и стиль | + Мощные средства для создания современных и стильных интерфейсов | - Ограниченные возможности по сравнению с современными GUI фреймворками |

**Продолжение таблицы 1.1**

|  |  |  |
| --- | --- | --- |
| Масштабируемость | + Легко масштабируется для крупных и сложных проектов | - Подходит для создания малых и средних приложений |
| Поддержка графики | |  | | --- | | + Широкие возможности для работы с графикой и анимацией |  |  | | --- | |  | | - Ограниченная поддержка современных графических возможностей |
| Модульность и расширяемость | + Высокая модульность, легко расширяем с помощью плагинов и модулей | + Поддерживает создание модульных приложений |

В ходе анализа двух GUI было решено выбрать QT Creator. Во-первых, Qt имеет отличную документацию и активное сообщество, что упрощает процесс обучения для разработчиков. Во-вторых, широкая документация и поддержка от сообщества обеспечивают доступ к полезным ресурсам и советам, что ускоряет разработку и решение возникающих проблем.

Кроме того, Qt предоставляет мощные средства для создания современных и стильных интерфейсов, превосходящих возможности WinForms, что является важным для привлекательности и удобства использования приложения. Qt также легко масштабируется для крупных и сложных проектов, что делает его идеальным для долгосрочных разработок. Наконец, широкие возможности для работы с графикой и анимацией делают Qt предпочтительным для приложений с интенсивной графикой, обеспечивая более высокое качество визуальных элементов. Эти преимущества делают Qt оптимальным выбором для нашего проекта.

# 2 программное проектирование

## **2.1 Функции клиента и сервера**

Клиент-серверное приложение "Платформа для проверки знаний" основано на взаимодействии между клиентской и серверной частями, каждая из которых выполняет свои функции для обеспечения корректной и эффективной работы системы.

Сервер играет ключевую роль в предоставлении функциональности приложения, обеспечивая соединение для клиентов и управляя всей серверной логикой. Клиентская часть приложения предоставляет пользователю интерфейс для взаимодействия с системой тестирования и осуществляет все операции, связанные с проведением и оценкой тестов.

В таблице 2.1 представлены функции сервера и клиента в приложении "Платформа для проверки знаний".

**Таблица 2.1** – Функции сервера и клиента

|  |  |  |
| --- | --- | --- |
| Функции | Сервер | Клиент |
| Подключение к серверу | **---** | Подключается к серверу для обмена данными. |
| Отправка запроса | Получает запросы от клиента. | Отправка запроса на получение вопроса из теста. |
| Обработка запросов | Реагирует на полученные запросы и обрабатывает их. | **---** |
| Отправка данных | Отправляет результат теста клиента. | Отправка результатов выполненного теста. |
| Обработка ошибок | Обрабатывает ошибки, возникающие при сетевых операциях. | **---** |
| Вывод отладочных сообщений | Выводит отладочные сообщения. | **---** |
| Получение данных | **---** | Получает результаты выполнения запросов от сервера. |

Диаграмма вариантов использования приложения представлена в Приложении А.

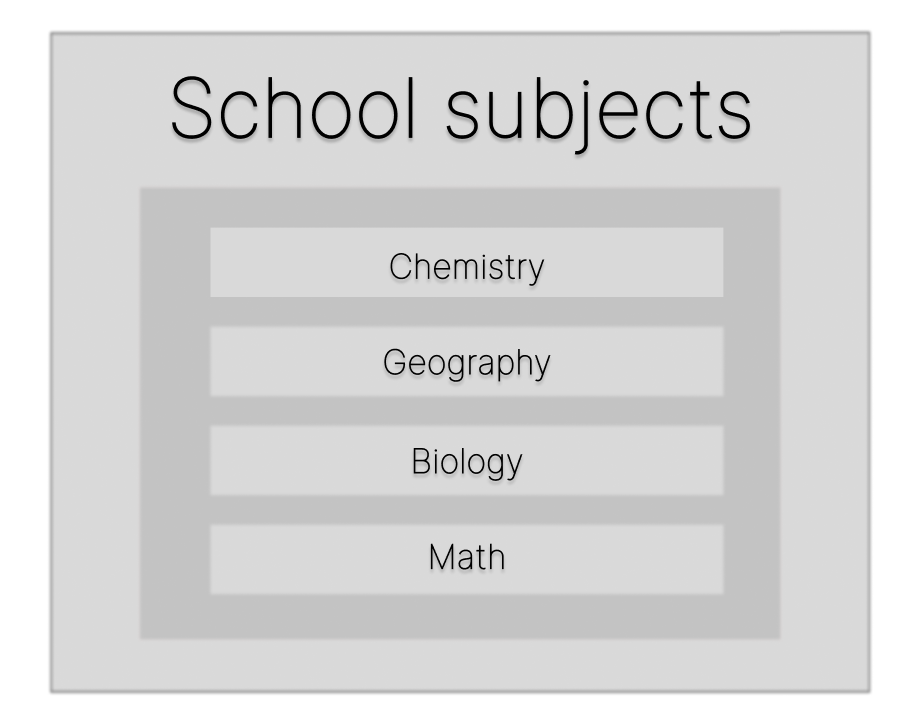
## **2.2 Макет пользовательского интерфейса**

Клиентская часть будет реализована в виде оконного приложения.

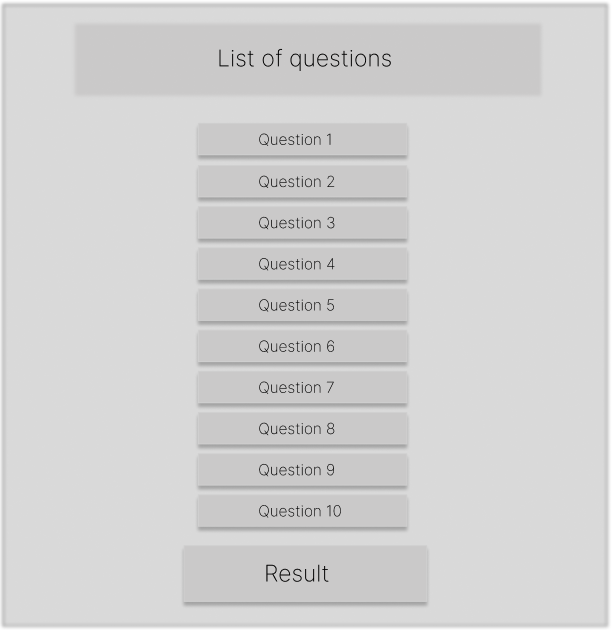
На рисунке 2.1 представлен макет окна при выборе предмета, по которому можно пройти тест. На представленном макете расположен заголовок окна и 4 кнопки позволяющие выбрать предмет по которому будет проходить тестирование.

На рисунке 2.2 представлен макет окна со списком вопросов. На макете располагается заголовок окна, кнопки для выбора и отображения вопросов и кнопка для проверки результатов теста.

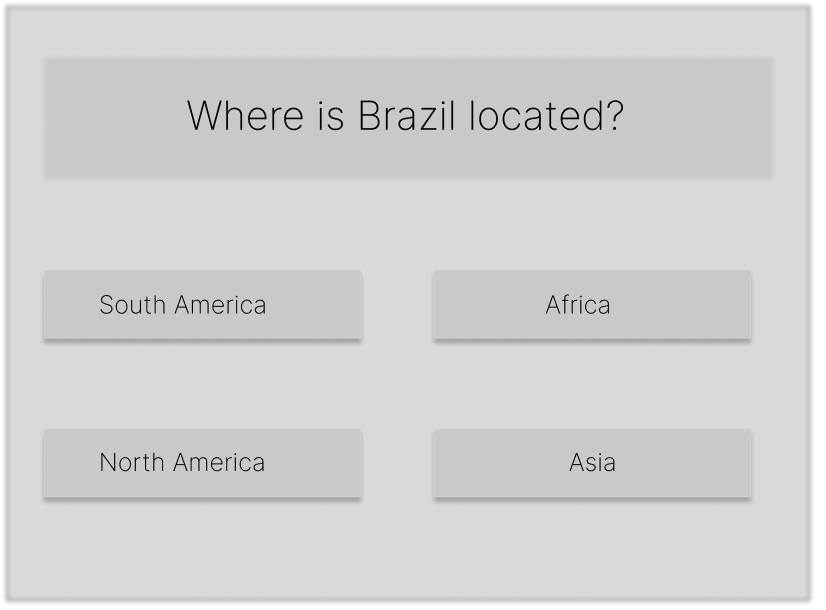
На рисунке 2.3 представлен макет с вопросом и вариантами ответов. На макете располагается вопрос и 4 кнопки для ответа на этот вопрос.



**Рисунок 2.1** – Окно выбора предметов



**Рисунок 2.2** – Окно списка вопросов



**Рисунок 2.3 –** Окно тестового вопроса с вариантами ответов

## **2.3 Прикладной протокол взаимодействия клиента и сервера**

В данной секции описывается процесс взаимодействия между клиентом и сервером с использованием сокетов. Сервер и клиент обмениваются информацией через заранее определённые запросы, отправляемые клиентом, и соответствующие ответы, предоставляемые сервером. Протокол взаимодействия между ними обеспечивает получение клиентом необходимых данных и обработку этих данных сервером.

Примеры запросов и их обработка сервером:

1. Запрос: GetGeoQuestion 1

Действия сервера:

- Сервер получает запрос и определяет, что клиент запросил первый географический вопрос.

- Сервер читает содержимое файла Question1.txt, содержащего первый

вопрос.

- Сервер отправляет содержимое файла клиенту.

1. Запрос: GetMathQuestion 1

Действия сервера:

- Сервер получает запрос и определяет, что клиент запросил первый математический вопрос.

- Сервер читает содержимое файла Question1.txt, содержащего первый вопрос.

- Сервер отправляет содержимое файла клиенту.

1. Запрос: Любой запрос, не соответствующий ожидаемому формату.

Действия сервера:

- Сервер получает запрос и определяет, что запрос некорректен.

- Сервер отправляет клиенту сообщение об ошибке Invalid command.

Таким образом, взаимодействие между клиентом и сервером осуществляется посредством отправки и получения строковых сообщений через TCP-соединение. Сервер обрабатывает запросы клиента, отправляет необходимые файлы или сообщения, и логирует все события для обеспечения надежности и отладки.

## **2.4 Алгоритм работы клиента и сервера**

Взаимодействие клиента и сервера в приложении "Платформа для проверки знаний" осуществляется через TCP-соединение, позволяя пользователям проходить тесты, получать доступ к вопросам и мгновенно видеть результаты, внесённые другими пользователями. Сервер обрабатывает запросы на получение вопросов и отправку ответов, логирует события и управляет соединениями с клиентами, обеспечивая стабильную и надёжную работу системы. Пользователи могут эффективно выполнять тестирование и получать результаты в режиме реального времени.

Диаграмма взаимодействий представлена в Приложении Б

# 3 Программная Реализация и тестирование

## **3.1 Детальная реализация функциональных частей программного обеспечения**

Согласно заданию по курсовому проектированию по дисциплине «Операционные системы и системные программы», разработано приложение «Платформа для проверки знаний», на языке программирования C++. Компоненты использованные при разработке описаны в диаграмме компонентов.(Приложение В)

### **3.1.1 Реализация функциональных частей клиента и сервера**

Для работы приложения были реализованы функции для взаимодействия клиента и сервера. Функция logMessage(), описанная в листинге 3.1, начинает работу сервера.

**Листинг 3.1 –** Функция startServer():

1. void logMessage(const std::string& message) {
2. std::ofstream logFile("server\_log.txt", std::ios::app);
3. if (logFile.is\_open()) {
4. std::time\_t currentTime;
5. time(&currentTime);
6. struct tm timeInfo;
7. localtime\_s(&timeInfo, &currentTime);
8. char buffer[20];
9. strftime(buffer, sizeof(buffer), "%Y-%m-%d %H:%M:%S", &timeInfo);
10. logFile << "[" << buffer << "] " << message << std::endl;
11. logFile.close();
12. }
13. else {
14. std::cerr << "Failed to open log file for writing." << std::endl;
15. }
16. }

Эта функция принимает строку сообщения и записывает ее в лог-файл server\_log.txt, добавляя текущее время.

**Листинг 3.2 –** Функция saveAnswer():

1. void saveAnswer(const std::string& subject, const std::string& questionNumber, const std::string& answer) {
2. std::string filename = "C:\\Users\\chete\\OneDrive\\Рабочий стол\\build-OSiSPkursach1-Desktop\_Qt\_6\_6\_2\_MinGW\_64\_bit-Debug\\Server2\\Server2\\GeoTest";
3. std::ofstream answerFile(filename, std::ios::app);
4. if (answerFile.is\_open()) {
5. answerFile << subject << ":" << questionNumber << ":" << answer << std::endl;
6. answerFile.close();
7. logMessage("Answer saved: " + subject + " - Question " + questionNumber + " - " + answer);
8. }

**Продолжение листинга 3.2**

1. else {
2. logMessage("Failed to open answer file: " + filename);
3. }

Эта функция сохраняет ответы в файл GeoTest в указанном формате

**Листинг 3.3 –** Функция sendFile():

1. void sendFile(SOCKET clientSocket, const std::string& filename) {
2. std::ifstream file(filename);
3. if (!file.is\_open()) {
4. logMessage("Failed to open file: " + filename);
5. std::string errorMsg = "Failed to open file: " + filename;
6. send(clientSocket, errorMsg.c\_str(), errorMsg.size(), 0);
7. return;
8. }
9. std::string line;
10. std::string fileContents;
11. while (std::getline(file, line)) {
12. fileContents += line + "\n";
13. }
14. int bytesSent = send(clientSocket, fileContents.c\_str(), fileContents.size(), 0);
15. if (bytesSent == SOCKET\_ERROR) {
16. logMessage("Failed to send file data: " + filename);
17. }
18. else {
19. logMessage("File sent successfully: " + filename);
20. }
21. file.close();

Эта функция отправляет содержимое файла клиенту через сокет.

**Листинг 3.4 –** Инициализация Winsock:

1. WSADATA wsaData;
2. int result = WSAStartup(MAKEWORD(2, 2), &wsaData);
3. if (result != 0) {
4. std::cerr << "WSAStartup failed: " << result << std::endl;
5. logMessage("WSAStartup failed with error: " + std::to\_string(result));
6. return 1;
7. }
8. SOCKET serverSocket = socket(AF\_INET, SOCK\_STREAM, IPPROTO\_TCP);
9. if (serverSocket == INVALID\_SOCKET) {
10. std::cerr << "Error creating socket: " << WSAGetLastError() << std::endl;
11. logMessage("Failed to create socket with error: " + std::to\_string(WSAGetLastError()));
12. WSACleanup();
13. return 1;
14. }

Этот блок кода инициализирует библиотеку Winsock с помощью функции WSAStartup(). Если инициализация прошла успешно, то возвращается 0, иначе код ошибки.

**Листинг 3.5 –** Создание серверного сокета и его привязка к адресу и порту:

1. *vSOCKET serverSocket = socket(AF\_INET, SOCK\_STREAM, IPPROTO\_TCP);*
2. *if (serverSocket == INVALID\_SOCKET) {*
3. *// ...*
4. *}*
5. sockaddr\_in serverAddr; serverAddr.sin\_family = AF\_INET;
6. serverAddr.sin\_addr.s\_addr = INADDR\_ANY; serverAddr.sin\_port = htons(12345);
7. if (bind(serverSocket, reinterpret\_cast<sockaddr\*>(&serverAddr),
8. sizeof(serverAddr)) == SOCKET\_ERROR) { // ... }

Этот блок кода связывает сокет с адресом и портом. Если привязка не удалась, программа завершает работу.

**Листинг 3.6 –** Начало прослушивания порта и Цикл обработки подключений:

1. if (listen(serverSocket, SOMAXCONN) == SOCKET\_ERROR) {
2. std::cerr << "Listen failed: " << WSAGetLastError() << std::endl;
3. logMessage("Listen failed with error: " + std::to\_string(WSAGetLastError()));
4. closesocket(serverSocket);
5. WSACleanup();
6. return 1;
7. }
8. std::cout << "Server listening on port 12345..." << std::endl;
9. logMessage("Server started and listening on port 12345.");
10. while (true) {
11. SOCKET clientSocket = accept(serverSocket, nullptr, nullptr);
12. if (clientSocket == INVALID\_SOCKET) {
13. std::cerr << "Accept failed: " << WSAGetLastError() << std::endl;
14. logMessage("Accept failed with error: " + std::to\_string(WSAGetLastError()));
15. continue;
16. }
17. // Отладочное сообщение при подключении пользователя
18. std::cout << "Client connected. Socket: " << clientSocket << std::endl;
19. logMessage("Client connected. Socket: " + std::to\_string(clientSocket));
20. char buffer[1024];
21. int bytesReceived = recv(clientSocket, buffer, sizeof(buffer), 0);
22. if (bytesReceived <= 0) {
23. std::cerr << "Recv failed: " << WSAGetLastError() << std::endl;
24. logMessage("Recv failed with error: " + std::to\_string(WSAGetLastError()));
25. closesocket(clientSocket);
26. continue;

Здесь сервер начинает прослушивание порта с помощью функции listen(). Если происходит ошибка, программа завершает работу.

Этот бесконечный цикл ожидает подключений клиентов с помощью функции accept(). Как только клиент подключается, создается сокет для общения с ним, и программа переходит к обработке запросов.

Это основные шаги и функции, реализованные в представленном коде. Разделение кода на логические блоки и комментарии делают его более понятным и легко читаемым.

**Листинг 3.7 –** Обработка запроса

1. if (request == "GetGeoQuestion 1") {
2. // Отладочное сообщение о запросе на получение вопроса 1
3. std::cout << "Sending question 1." << std::endl;
4. std::string filename = "C:\\Users\\chete\\OneDrive\\Рабочий стол\\build-OSiSPkursach1-Desktop\_Qt\_6\_6\_2\_MinGW\_64\_bit-Debug\\Server2\\Server2\\GeoTest\\Question1.txt";
5. std::cout << "Sending question from file: " << filename << std::endl;
6. sendFile(clientSocket, filename);
7. }

### Сервер ждёт запрос и обрабатывает если он ему знаком

### **Листинг 3.8 -** Закрытие сокета

1. closesocket(serverSocket);
2. WSACleanup();
3. return 0;
4. }

### Закрытие сокета и прекращение работы сервера

### **3.1.2 Реализация пользовательского интерфейса клиента**

Реализация пользовательского интерфейса предполагает создание окна и добавление кнопок для взаимодействия пользователя с приложением.  
 Функция Tests(), приведённая в листинге 3.9ы, создаёт главное окно приложения.

**Листинг 3.9 –** Окно Test():

1. #include "tests.h"
2. #include "ui\_tests.h"
3. #include "geotests.h"
4. #include "geoquestlist.h"
5. Tests::Tests(QWidget \*parent)
6. : QDialog(parent)
7. , ui(new Ui::Tests)
8. {
9. ui->setupUi(this);
10. }
11. Tests::~Tests()
12. {
13. delete ui;
14. }
15. void Tests::on\_pushButton\_clicked()

**Продолжение листинга 3.9**

1. {
2. // Создаем экземпляр MainWindow динамически
3. GeoQuestList \*w = new GeoQuestList();
4. // Отображаем окно MainWindow
5. w->show();
6. }

.Начальное окно созданое для выбора предмета для теста.

**Листинг 3.10 –** Функция compareAnswers():

1. int GeoQuestList::compareAnswers(const QString &answersForTestFile, const QString &userAnswersFile) {
2. QFile testFile(answersForTestFile);
3. QFile userFile(userAnswersFile);
4. if (!testFile.open(QIODevice::ReadOnly | QIODevice::Text)) {
5. qDebug() << "Failed to open test answers file for reading.";
6. return -1;
7. }
8. if (!userFile.open(QIODevice::ReadOnly | QIODevice::Text)) {
9. qDebug() << "Failed to open user answers file for reading.";
10. return -1;
11. }
12. QTextStream testStream(&testFile);
13. QTextStream userStream(&userFile);
14. int score = 0;
15. int questionNumber = 0;
16. while (!testStream.atEnd() && !userStream.atEnd()) {
17. QString testLine = testStream.readLine();
18. QString userLine = userStream.readLine();
19. QStringList testParts = testLine.split(", Answer: ");
20. QStringList userParts = userLine.split(", Answer: ");
21. if (testParts.size() != 2 || userParts.size() != 2) {
22. qDebug() << "Invalid format in one of the files.";
23. return -1;
24. }
25. int testAnswer = testParts[1].trimmed().toInt();
26. int userAnswer = userParts[1].trimmed().toInt();
27. if (testAnswer == userAnswer) {
28. ++score;
29. }
30. ++questionNumber;
31. }
32. testFile.close();
33. userFile.close();
34. return score;
35. }
36. void GeoQuestList::onQuestionButtonClicked()
37. {
38. // Определение номера вопроса в зависимости от нажатой кнопки
39. int questionNumber = 0;

**Продолжения листинга 3.10**

1. if (sender() == ui->Question1Btn) {
2. questionNumber = 1;
3. }
4. else if (sender() == ui->Question2Btn) {
5. questionNumber = 2;
6. } // Добавьте аналогичные проверки для остальных кнопок
7. else if (sender() == ui->Question3Btn) {
8. questionNumber = 3;
9. }
10. else if (sender() == ui->Question4Btn) {
11. questionNumber = 4;
12. }
13. else if (sender() == ui->Question5Btn) {
14. questionNumber = 5;
15. } // Добавьте аналогичные проверки для остальных кнопок
16. else if (sender() == ui->Question6Btn) {
17. questionNumber = 6;
18. }
19. else if (sender() == ui->Question7Btn) {
20. questionNumber = 7;
21. }
22. else if (sender() == ui->Question8Btn) {
23. questionNumber = 8;
24. } // Добавьте аналогичные проверки для остальных кнопок
25. else if (sender() == ui->Question9Btn) {
26. questionNumber = 9;
27. }
28. else if (sender() == ui->Question2Btn10) {
29. questionNumber = 10;
30. }
31. // Создание и открытие окна GeoQuest1 с передачей номера вопроса
32. GeoQuest1 \*geoQuest1 = new GeoQuest1(this);
33. geoQuest1->setQuestionNumber(questionNumber);
34. geoQuest1->show();
35. }

Функция compareAnswers:

1. Открывает два файла с тестовыми и пользовательскими ответами.
2. Сравнивает строки из обоих файлов.
3. Разбивает строки на части, извлекая номер вопроса и ответ.
4. Сравнивает ответы и увеличивает счёт, если они совпадают.Возвращает итоговый счёт.
5. Функция onQuestionButtonClicked:
6. Определяет номер вопроса в зависимости от нажатой кнопки.
7. Создаёт и открывает окно GeoQuest1, передавая номер вопроса

**Листинг 3.11 –** Функция on\_ResultBtn\_clicked():

1. *void GeoQuestList::on\_ResultBtn\_clicked()*
2. *{*
3. *QString testAnswersFile = "C:/Users/chete/OneDrive/Рабочий стол/build-OSiSPkursach1-Desktop\_Qt\_6\_6\_2\_MinGW\_64\_bit-Debug/Server2/Server2/GeoTest/answersForTest.txt";*
4. *QString userAnswersFile = "C:/Users/chete/OneDrive/Dokumenty/OSiSPkursach1/build/Desktop\_x86\_windows\_msvc2022\_pe\_64bit-Debug/answers.txt"*
5. *int score = compareAnswers(testAnswersFile, userAnswersFile);*
6. *qDebug() << "User score:" << score;*
7. *}*

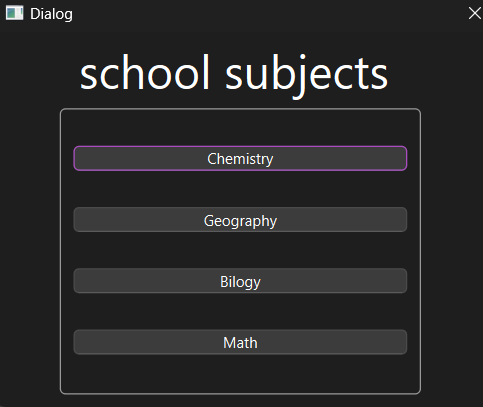
Эта функция открывает файлы с ответами на тест как для теста, так и для пользователя, а затем вызывает функцию compareAnswers для сравнения ответов и выводит результат пользовательского счета в отладочную консоль.

**Листинг 3.12** - функция saveAnswerToFile()

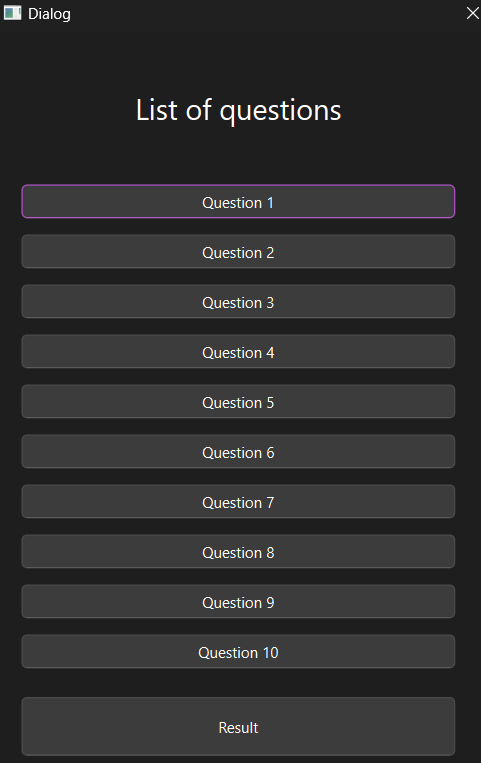
1. void GeoQuest1::saveAnswerToFile(int questionNumber, const QString &answer) {
2. QFile file("answers.txt"); // Используем более простой путь
3. if (file.open(QIODevice::WriteOnly | QIODevice::Append | QIODevice::Text)) {
4. QTextStream out(&file);
5. out << "Question: " << questionNumber << ", Answer: " << answer << "\n";
6. file.close();
7. } else {
8. qDebug() << "Failed to open file for writing.";
9. }
10. }

Функция saveAnswerToFile() сохраняет ответ в файл answers.txt

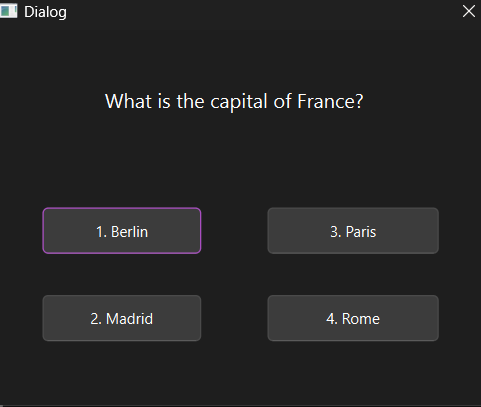
Qt Creator также включает в себя утилиту Qt Designer, что позволяет обработать внешний вид окна приложения, добавляя и перетаскивая элементы. Процесс работы и создание пользовательского интерфейса с помощью утилиты Qt Designer представлен на рисунке 3.1.



**Рисунок 3.1 –** Пользовательский интерфейс.



**Рисунок 3.2 –** Пользовательский интерфейс.



**Рисунок 3.3 –** Пользовательский интерфейс.

Разработка интерфейса упрощает использование приложения пользователем. Упрощается навигация по приложению. Интерфейс интуитивно понятен.

## **Сопроводительная документация программного обеспечения**

### **3.2.1 Описание установки и запуска приложения**

Для установки приложения требуется поместить папку с приложением клиента на ваше устройство и изменить путь для создания файла конфигурации сетевого адреса. Папку с приложением сервера поместить на устройство-сервер. Изначально файлы, для их дальнейшего использования на сервере, могут быть помещены на диск на том устройстве, которое выступает в качестве сервера.

Для запуска сервера и клиента требуется запустить соответствующие проекты. Чтобы установить соединение, требуется запустить сервер и клиент и убедиться в том, что соединение установлено

### **3.2.2 Руководства пользователя**

Необходимым требованием для совместной работы нескольких пользователей является нахождение их персональных компьютеров в одной локальной сети. В приложении можно взаимодействовать с файлами формата «.txt». При запуске клиента пользователь должен выбрать тест и попытаться получить ответ.

## **Тестирование прогаммного обеспечения**

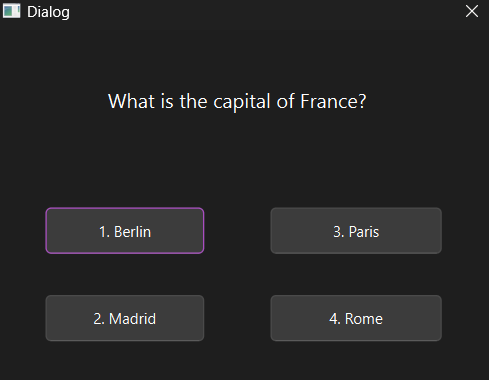
В данном разделе будет производится тестирование функций взаимодействия клиента и сервера. Результаты тестирования приведены в таблице 3.1.

**Таблица 3.1** – Тестирование функций приложения

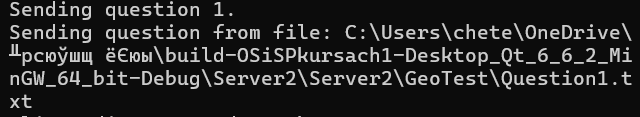
|  |  |  |  |
| --- | --- | --- | --- |
| **Номер теста** | **Название теста** | **Шаги вопроизведения** | **Ожидаемый результат** |
| 1 | Запуск сервера | 1. Открыть(“Server2.exe”) | Сообщение об запуске сервера и готовность подключения к нему (рисунок 3.4) |
| 2 | Запрос вопроса у сервера | 1.Открыть(“Server2.exe) 2.Открыть клиент-серверное приложение “Платформа для проверки знаний)  3.Выбор теста  4.Выбор любого вопроса | Получение вопроса со стороны сервера в специально созданное окно(рисунок 3.5) и получение отладочного сообщения на сервер об отправке файла(рисунок 3.6) |
| 3 | Запись ответа клиента в специальный файл | 1.Открыть(“Server2.exe) 2.Открыть клиент-серверное приложение “Платформа для проверки знаний)  3.Выбор теста  4.Выбор любого вопроса  5. Нажатие на кнопку ответа | Запись данных в файл и получение отладочного сообщения в QT creator(рисунок 3.7) |
| 4 | Проверка правильности ответов на тест | 1.Открыть(“Server2.exe) 2.Открыть клиент-серверное приложение “Платформа для проверки знаний)  3.Выбор теста  4.Решение теста  5. Нажатие на кнопку | Сообщение об получении баллов за тест (рисунок 3.8) |



**Рисунок 3.4 –** Запуск и готовность сервера к подключению



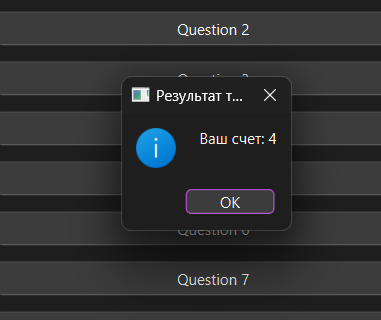
**Рисунок 3.5 –** Получение вопроса со стороны сервера



**Рисунок 3.6 –** Получение отладочного сообщения на сервере



**Рисунок 3.7 –** Получение отладочного сообщения в QT creator



**Рисунок 3.8 –** Получение баллов в QT creator

**Таблица 3.2** - Результат выполнения тестов

|  |  |  |
| --- | --- | --- |
| **Номер теста** | **Название теста** | **Результат тестирования** |
| 1 | Запуск сервера | Успешно |
| 2 | Запрос вопроса у сервера | Успешно |
| 3 | Запись ответа клиента в специальный файл | Успешно |
| 4 | Проверка правильности ответов на тест | Успешно |

Исходя из результатов тестирования ошибок в работе программы выявлено не было. Можно сделать вывод, что приложение готово к использованию.

# Заключение

Разработанное клиент-серверное приложение "Система онлайн проверки знаний" представляет собой надежное и безопасное программное обеспечение, созданное с использованием среды разработки Qt Creator. Приложение обеспечивает пользователям возможность подключаться к серверу для прохождения тестов, получения вопросов и отправки ответов как через интернет, так и через локальную сеть.

Важными аспектами приложения являются его защищенность и возможность использования в локальной сети для проверки студентов/учеников. Оно предоставляет уверенность в сохранности данных и конфиденциальности пользователей. Реализация функций сервера, клиента и тестирования обеспечивает безопасное и стабильное взаимодействие, а проверка корректности введенных данных гарантирует надежность работы системы.

Полное тестирование приложения позволяет убедиться в его соответствии требованиям и стабильности работы. Результаты разработки обеспечивают мгновенный доступ к вопросам и ответам, а также гарантируют актуальность данных для всех пользователей. В целом, приложение эффективно решает поставленные задачи и может быть успешно применено для проверки знаний студентами или другими пользователями в защищенной и удобной среде.

# Список использованных источников

1. The TCP/IP Guide [Электронный ресурс]. - Режим доступа: http://www.tcpipguide.com/. - Дата доступа: 13.04.2024.

2. Google Forms [Электронный ресурс]. - Режим доступа: https://www.google.by/intl/ru/forms/about/. - Дата доступа: 18.04.2024.

3. Moodle - Open-source learning platform [Электронный ресурс]. - Режим доступа: https://moodle.org/. - Дата доступа: 18.04.2024.

4. Quizizz | Free Online Quizzes, Lessons, Activities and Homework [Электронный ресурс]. - Режим доступа: https://quizizz.com/?lng=ru. - Дата доступа: 18.04.2024.

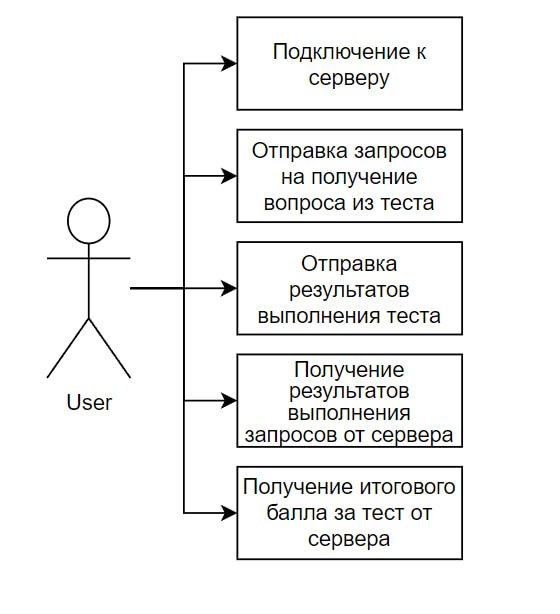
5. Qt Creator Manual [Электронный ресурс]. - Режим доступа: https://doc.qt.io/qtcreator/. - Дата доступа: 23.05.2024.

6. Windows Forms for .NET documentation | Microsoft Learn [Электронный ресурс]. - Режим доступа: https://learn.microsoft.com/en-us/dotnet/desktop/winforms/?view=netdesktop-6.0. - Дата доступа: 23.05.2024.

7. The Internet Engineering Task Force (IETF) [Электронный ресурс]. - Режим доступа: https://www.ietf.org/. - Дата доступа: 23.05.2024.

# Приложение А

**Диаграмма вариантов использования.**



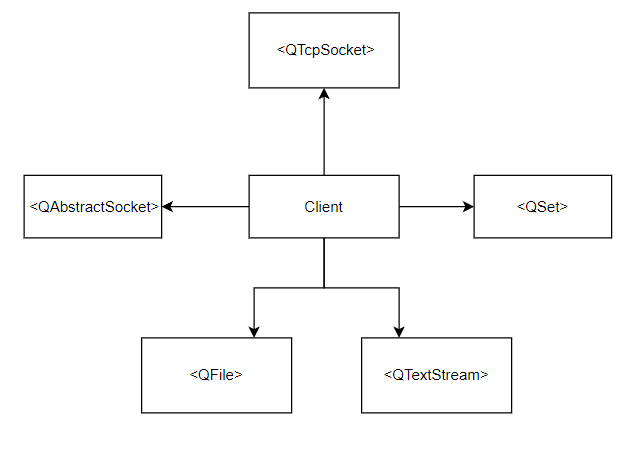
# Приложение Б

**Диаграмма взаимодействий.**

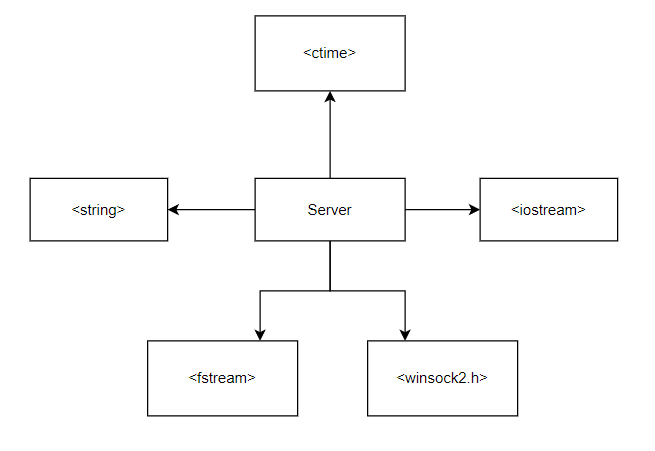


# Приложение В

**Диаграмма компонентов.**



**Рисунок В.1** – Диаграмма компонентов клиента



**Рисунок В.2 –** Диаграмма компонентов сервера