Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт
Вычислительная техника
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой
_____ _____
подпись инициалы, фамилия
«____» _____ 20 __ г.

БАКАЛАВРСКАЯ РАБОТА

09.03.01 Информатика и вычислительная техника
код и наименование направления

Редактор файлов формата GIFT
тема

Пояснительная записка

Руководитель	_____	_____	Д.А. Швец
	подпись, дата	должность, ученая степень	инициалы, фамилия
Выпускник	_____		Д.А. Коробкин
	подпись, дата		инициалы, фамилия
Нормоконтролер	_____	_____	Д.А Швец
	подпись, дата	должность, ученая степень	инициалы, фамилия

Красноярск 2021

Министерство науки и высшего образования РФ
Федеральное государственное автономное образовательное учреждение высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Космических и информационных технологий
институт
Вычислительная техника
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой

_____ _____
подпись инициалы, фамилия
«____» _____ 20 __ г.

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
в форме бакалаврской работы

Студенту Коробкин Дмитрий Алексеевич
фамилия, имя, отчество

Группа КИ17-06Б Направление (специальность) 09.03.01
номер код

Информатика и вычислительная техника
наименование

Тема выпускной квалификационной работы Редактор файлов формата GIFT

Утверждена приказом по университету № _____ от _____

Руководитель ВКР Д.А. Швец, канд. техн. наук, доцент, кафедра ИВТ
инициалы, фамилия, должность, ученое звание и место работы

Исходные данные для ВКР: разработать кроссплатформенное приложение для редактирования файлов формата GIFT.

Перечень разделов ВКР: анализ предметной области, проектирование приложения, реализация приложения.

Перечень графического материала: презентация в формате PowerPoint.

Руководитель ВКР _____ Д.А. Швец
подпись инициалы и фамилия

Задание принял к исполнению _____ Д.А. Коробкин
подпись инициалы и фамилия студента

« ___ » _____ 20__ г.

РЕФЕРАТ

Выпускная квалификационная работа на тему «Редактор файлов формата GIFT» содержит 41 страницу текстового документа, 26 иллюстраций, 1 таблицу и 14 использованных источников.

КРОССПЛАТФОРМЕННОЕ ПРИЛОЖЕНИЕ, РЕДАКТОР ФАЙЛОВ, ВОПРОСЫ, GIFT, ОС WINDOWS, ОС MACOS, ОС LINUX.

Цель выпускной квалификационной работы – разработка кроссплатформенного приложения для редактирования файлов формата GIFT.

В результате выполнения выпускной квалификационной работы была проанализирована предметная область по теме и определена структура приложения. В ходе разработки приложения были реализованы следующие функции: добавление и удаление вопросов из файла, создание и сохранение новых GIFT-файлов и редактирование вопросов в экспортированных файлах.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Системы электронного обучения всё чаще интегрируются в образовательный процесс. Эффективность таких систем очевидна: преподаватели выстраивают в них свою модель обучения, а студенты получают постоянный доступ к учебным материалам. Преподаватели могут загружать в курсы все необходимые материалы, составлять задания, смотреть за успеваемостью студентов, держать связь с каждым учащимся. С помощью систем электронного обучения учебный процесс становится более структурированным и всеобъемлющим. На сегодняшний день удаленное образование широко используется во многих странах, так как доступ в интернет есть практически у каждого человека.

Существует много различных систем электронного обучения, и каждое учебное заведение самостоятельно выбирает платформу для удаленного обучения, в зависимости от своих потребностей. Одной из самых популярных платформ является Moodle [1]. Moodle – система с открытым исходным кодом, функционал и дизайн которой может изменяться и настраиваться с помощью плагинов [1].

Одной из функций Moodle является поддержка функционала онлайн тестов. Электронное тестирование автоматизирует процесс проверки результатов и исключает возможность ошибки. Платформа Moodle предоставляет несколько способов реализации онлайн-тестирования, один из них – файлы формата GIFT. Данный формат позволяет локально создавать тесты, экспортировать и импортировать их в электронные курсы. Для того, чтобы платформа правильно считывала вопросы, написанные в файле, необходимо соблюдать множество правил. Таким образом, для того, чтобы правильно составить тест, преподаватель вручную прописывает спецсимволы для каждого вопроса. Для того, чтобы сократить время на составление тестов, а также упростить процесс их создания необходим редактор, который

поддерживает сложный синтаксис GIFT-файлов. Существующие текстовые редакторы не могут полностью решить проблемы, так как они не позволяют автоматизировать процесс создания тестов.

Целью данной работы является создание приложение для редактирования файлов формата GIFT. Для достижения поставленной цели необходимо выполнить следующие задачи:

- Провести анализ предметной области;
- Определить функциональные требования к приложению;
- Определить структуру приложения;
- Спроектировать приложение;
- Разработать кроссплатформенное приложение;

1 Анализ предметной области

1.1 Система электронного обучения Moodle

Электронное обучение – это управление процессом обучения и передача знаний с помощью технологий. В данном формате обучения широко используются информационные и телекоммуникационные технологии, преимущественно Интернет, а также корпоративные сети компаний, или даже компакт-диски [1].

Системы управления обучением позволяют разрабатывать, хранить, управлять и распространять учебные материалы, а также позволяют получить к ним одновременный доступ большому количеству людей.

Moodle — это бесплатное, открытое веб-приложение, система электронного обучения. На базе Moodle можно создать собственную специализированную платформу для обучения студентов или сотрудников [2].

Moodle представляет собой бесплатную систему электронного обучения. Она является одной из самых известных платформ для электронного образования, переведена на множество языков, а также используется в самых крупных и известных институтах во всем мире. Это открытое веб-приложение, с помощью которого можно реализовать собственную уникальную среду для обучения студентов или сотрудников [2].

1.1.2 Тесты в Moodle

Для того, чтобы создать тест в платформе Moodle, необходимо использовать элемент курса «Тест». С помощью настроек можно изменить параметры теста: установить ограниченное количество попыток прохождения, сделать ограничение по времени, отсортировать вопросы по порядку, или наоборот, перемешивать их, включить добавление случайных вопросов, которые будут выбираться автоматически из банка вопросов. Вопросы, которые

могут быть добавлены в тест для удобства разделены на типы: множественный выбор, верно/неверно, на соответствие, короткий ответ, числовой ответ [3].

После прохождения теста оценка выставляется автоматически. Исключением являются вопросы типа «Эссе», так как ответом на данный вопрос является текст, написанный самим студентом, и требует личной проверки преподавателя. После того, как будет произведена проверка теста, оценка за тестирование автоматически переносится в журнал оценок. С помощью дополнительных функций преподаватель может добавить подсказки, а также выбрать, будут ли правильные ответы показаны или скрыты для студента.

Электронное тестирование можно использовать для любой дисциплины. Они могут использоваться для самопроверки, или для проверки пройденного материала, в качестве финального экзамена или для моментальной проверки освоенной темы. Также можно составить итоговое тестирование, используя вопросы из промежуточных тестирований.

1.1.3 Банк вопросов Moodle

Банк вопросов – функция, которая позволяет учителю создавать, просматривать и редактировать вопросы в базе данных категорий вопросов [6].

Добавить вопросы в банк вопросов можно не только импортом заранее созданного файла, можно также создать через редактор банка вопросов (Рисунок 1). Вопросы сгруппированы по категориям. Таким образом, с помощью банка вопросов учитель сам выбирает какие вопросы он добавит в тот или иной тест.

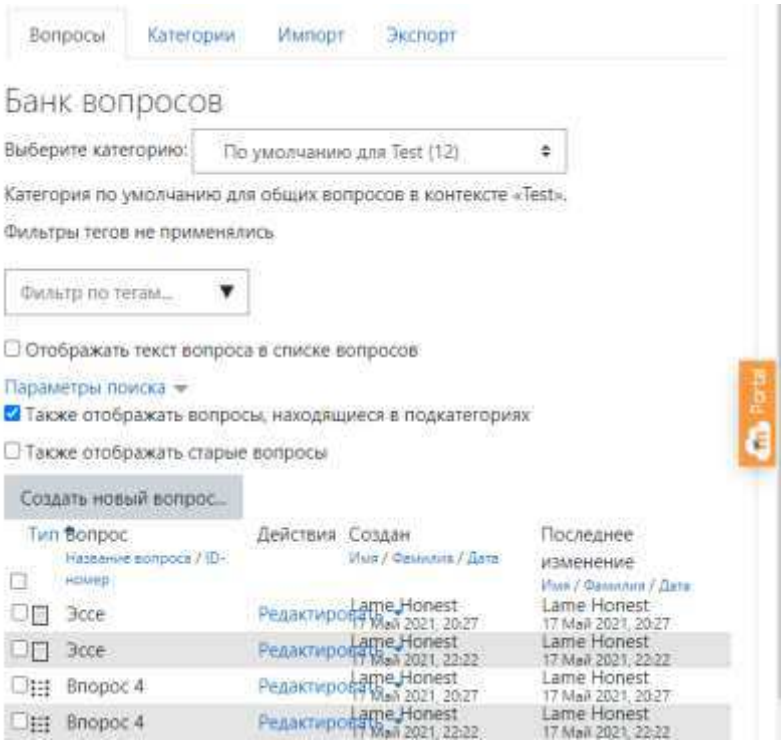

Рисунок 1 – Банк вопросов

1.1.4 Форматы файлов для импорта в банк вопросов

Moodle имеет несколько различных файловых форматов, которые можно использовать для импорта вопросов (Рисунок 2) [7]:

«Вложенные ответы» (Cloze) – вопросы состоят из текста (в формате Moodle), в который встроены различные ответы: числовые ответы, короткие ответы, множественный выбор.

Формат «Пропущенное слово» - позволяет импортировать вопросы типа «Пропущенное слово» из текстового файла.

Формат GIFT – позволяет пользователю создавать вопросы с различными ответами: краткий ответ, вопрос с несколькими вариантами ответов, истинно-ложный, вопрос на соответствие и числовой вопрос. Можно использовать текстовый редактор для написания вопросов в простом формате, который можно импортировать. Формат GIFT также является форматом файла экспорта, доступным в банке вопросов.

Формат Aiken – позволяет создать вопросы с несколькими вариантами ответов в текстовом файле. Формат GIFT имеет гораздо больше возможностей

и менее подвержен ошибкам, но обладает более сложной структурой, чем Aiken.

Формат Blackboard – позволяет импортировать вопросы из ZIP-файла, сохранённого при экспорте в формате Blackboard. Поддерживается импорт изображений из ZIP-файла.

Формат Examview – позволяет импортировать вопросы из XML-файла Examview 4. Для более новых версий может использоваться формат Blackboard.

Формат Moodle XML – специальный формат для Moodle, позволяющий импортировать и экспортировать вопросы для модуля Викторина.

Формат WebCT – позволяет импортировать вопросы типа множественный выбор и короткий ответ из текстового файла в формате WebCT.

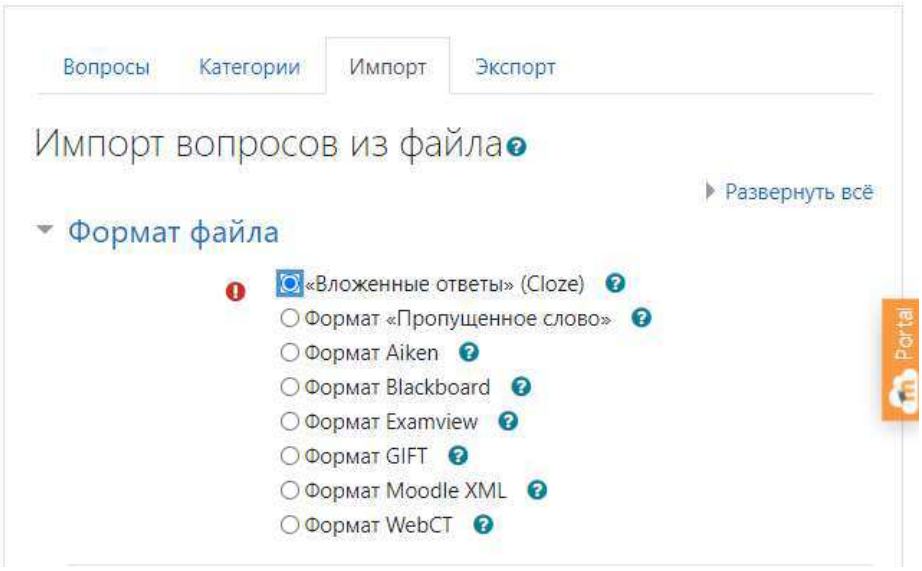

Рисунок 2 – Выбор формата файла

1.1.5 Экспорт вопросов

Вопросы можно экспортировать из банка вопросов в любом из 4 форматов:

- Aiken;

- GIFT;

- Moodle XML;

- XHTML;

Также вопросы можно экспортировать в формате Word, но для этого необходим дополнительный плагин.

1.1.6 Кодирование текста для импорта в Moodle

Набор символов - определённая таблица кодировки конечного множества знаков. Такая таблица сопоставляет каждому символу последовательность длиной в один или несколько байтов.

Для импорта файлов в банк вопросов следует использовать кодировку UTF- 8, которая является юникод-кодировкой переменной длинны, с помощью которой можно представить любой символ юникода [5].

1.2 Постановка задачи

Одним из недостатков экспорта вопросов в Moodle является перечень правил, которые необходимо соблюдать при создании вопросов в формате GIFT. Целью текущей бакалаврской работы является разработка мобильного приложения для работы с GIFT файлами. Разработанное приложение должно уметь работать с различными типами вопросов и файлами.

Согласно заданию, разрабатываемый редактор файлов формата GIFT должен соответствовать следующим требованиям:

- файлы, созданные в редакторе должны корректно отображаться в системе Moodle после импорта.

- файлы в формате GIFT, которые были экспортированы из системы Moodle должны корректно считываться редактором.

- кодировка файлов, созданных в редакторе должна быть UTF-8.

1.3 Функциональные возможности

- Возможность редактирования вопросов, находящихся в файле формата GIFT;
- Возможность создания и сохранения файлов;
- Возможность добавления новых вопросов в файл;
- Возможность удаления вопросов из файла;
- Возможность редактирования экспортированных GIFT файлов из системы Moodle;

1.4 Анализ существующих решений

На момент написания текущей ВКР аналогичных решений найдено не было. Для создания вопросов в GIFT файлах, пользователи используют различные текстовые редакторы, например, блокнот в ОС Windows или Microsoft Office Word. В ОС MacOS пользователи используют предустановленный текстовый редактор TextEdit (Рисунок 3).

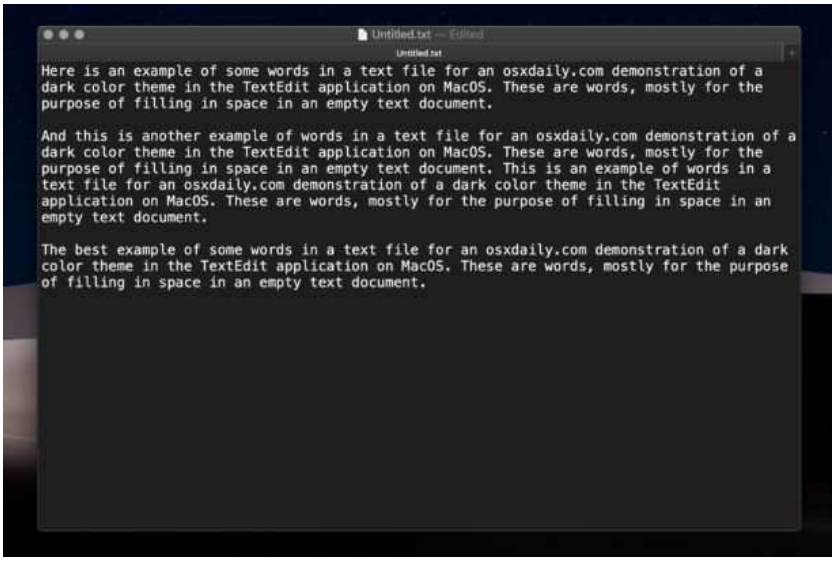

Рисунок 3 – Текстовый редактор TextEdit

2 Проектирование приложения

2.1 Формат файлов GIFT

Формат файла - спецификация структуры данных, записанных в компьютерном файле [8]. Идентификатор формата файла, как правило, указывается в конце имени файла в виде «расширения». Расширение имени файла помогает идентифицировать формат данных, содержащихся в файле, программам, которые могут с ним работать. Иногда формат данных дополнительно указывается в начале содержимого файла.

2.1.1 Структура файлов формата GIFT

В файлах формата GIFT может храниться большое количество вопросов и это предоставляет быстрый способ массовой загрузки вопросов в курс Moodle. Для того, чтобы создать файл со множеством вопросов, необходимо соблюдать одно условие: между вопросами должна быть хотя бы одна пустая строка.

Существует строгая инструкция, в которой, при создании вопросов сначала указывается текст вопроса, затем ответы. Опционально, перед текстом вопроса можно указать его заголовок. Каждый параметр вопроса должен разделяться специальными символами (Таблица 1).

Система Moodle может распознавать файлы формата GIFT только в кодировке UTF-8. Не допускается использование Unicode, так как Moodle не сможет распознать этот формат. Формат ANSI будет работать только для языков без каких-либо специальных символов [9].

При рассмотрении символов формата GIFT (Таблица 1), а также его структуры, можно заметить, что для каждого типа вопросов необходимо использовать определённый специальный символ.

Таблица 1 – Символы формата GIFT.

Символы	Использование
//text	Комментарий до конца строки (опционально)
::title::	Заголовок вопроса (опционально)
text	Текст вопроса (становится заголовком, если заголовок не указан)
[...format...]	Формат следующего фрагмента текста. Возможные варианты: [html], [moodle], [plain] и [markdown]. По умолчанию для текста вопроса используется значение [moodle], для других частей вопроса по умолчанию используется формат, используемый для текста вопроса
{	Начать ответ(ы)
}	Закончить ответ(ы)
{T} или {F}	Верный или ложный ответ; также {TRUE} и {FALSE}
{...=right...}	Правильный ответ при множественном выборе
{...~wrong...}	Неправильный ответ при множественном выборе или множественном ответе
{...=item-> match...}	Ответ на совпадающие вопросы

При экспорте вопросов из файла формата GIFT в курс Moodle специальные символы (Таблица 1) не могут быть использованы в тексте вопроса, так как это создаст конфликтную ситуацию при прохождении фильтра: спецсимволы используются для разделения частей вопросов. Однако очевидно, что избежать использования спецсимволов нельзя – они используются в формулах. Для того, чтобы решить данную проблему, можно использовать обратный слеш «\». Это будет «пропуском» для символов управления. Когда вопрос пройдет через обработку, обратный слеш удалится и не отобразится в Moodle.

В формате GIFT вопрос не может содержать пустых строк, а для разделения вопросов необходимо использовать между ними как минимум одну пустую строку.

Все вопросы оформляются по одному виду: ::Название вопроса:: Текст вопроса{ Варианты ответов }.

Для каждого вопроса можно добавить комментарии, если это необходимо, например, для оставления пометок для самого автора. При загрузке вопросов в Moodle комментарии не перенесутся. Если строка начинается с двойного обратного слеша (//), то текст, написанный после этого спецсимвола не будет учитываться фильтром.

2.1.2 Типы вопросов в GIFT файлах

Файлы формата GIFT поддерживают 7 типов вопросов, которые система Moodle может считать [10]:

1. Описание. Вопрос «описание» - информационная страница с текстом, который обычно помещается между отдельными вопросами, соответственно ответ на данный тип вопроса не требуется. Пример: ::Лекция 1::Данный тест содержит вопросы по лекции 1.

2. Множественный выбор. В качестве ответа на данный тип вопроса, студенту даются на выбор несколько вариантов. На данный вопрос может быть один или несколько правильных ответов. Для обозначения верных ответов используется символ (=), а неправильные ответы обозначаются символом (~). Если необходимо добавить несколько правильных ответов, для их обозначения используется символ (%), который взвешивает варианты. Пример: ::Вопрос 1:: Сфинкс это - : { ~порода собаки ~порода птицы ~порода кролика =порода кота }.

3. Краткий ответ. В качестве ответа на данный тип вопроса используется слово или короткая фраза. Можно добавить несколько правильных ответов, а также присвоить каждому ответу разную оценку. Введенный студентом ответ сравнивается с образцами ответов, в которых могут использоваться подстановочные знаки. Ответы в данных вопросах начинаются символом (=), так обозначается правильный ответ. Пример: ::Вопрос 1::В каком году началась Великая Отечественная Война?{ =в 1941 =в 1941 году }.

4. Верно-неверно. В качестве ответа на данный тип вопроса, студенту необходимо выбрать между вариантами «Верно» и «Неверно». Ответ может быть написан как {TRUE} или {FALSE}, или в сокращенном {T} или {F}. Пример: ::Вопрос 1::Столица Китая - Пекин. {TRUE}.

5. На соответствие. Для ответа на данный тип вопроса, студенту необходимо соотнести элементы из первой группы с элементами второй группы. Верные пары обозначаются знаком (=), для разделения используется знак (->). Для данного типа вопросов недоступно процентное оценивание. Для создания вопроса на соответствие необходимо минимум три совпадающие пары.. Пример: ::Название вопроса:: Укажите { =Алиса в стране чудес -> Безумный Шляпник =Золотая Рыбка -> Рыбка =Колобок -> Лиса }.

6. Числовой ответ. Для ответа на данный тип вопроса, студенту необходимо произвести вычисления. Для ответа можно добавить погрешность,

допустимую от верного ответа. При создании ответа, погрешность отделяется двоеточием . Для обозначения ответа используется символ (#). Например, если правильный ответ находится в диапазоне от 55.5 до 56.5, тогда вопрос должен быть написан так: {#56:0.5}. Если погрешность не определена, то по умолчанию она устанавливается в ноль. Пример: ::Вопрос 1:: Когда родилась Клара Цеткин (год)? {#1857}.

7. Эссе. Это единственный тип вопроса, который не проверяется автоматически. Данный тип вопроса содержит пустое поле для ответа, где студент описывает свой взгляд на заданную тему. В скобках ничего не нужно писать. Пример: ::Эссе::В чем смысл жизни? {}.

2.2 Определение функциональных требований

При рассмотрении структуры файлов формата GIFT, а также при детальном рассмотрении различных типов вопросов в Moodle, были сформированы следующие функциональные требования:

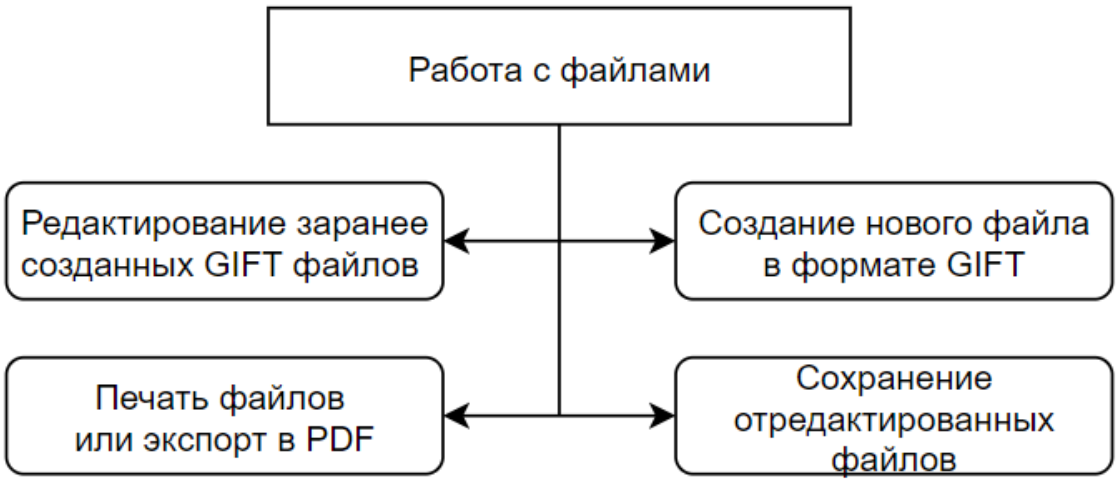

Рисунок 4 – Отображение функционала для работы с файлами

Модуль Работа с файлами разделяется на 4 подмодуля (Рисунок 4):

а) Редактирование заранее созданных файлов GIFT: это могут быть файлы, экспортированные из курса Moodle, или текстовые файлы, созданные пользователем.

б) Создание нового файла в формате GIFT, где созданный файл отвечает всем требованиям системы Moodle, который может импортироваться в курс.

в) Сохранение отредактированных файлов должно происходить в том каталоге, который выберет пользователь. Доступность каталога для записи и редактирования зависит от ОС и прав пользователя.

г) Печать файлов подразумевает распечатку текста файла формата GIFT на доступном принтере в ОС.

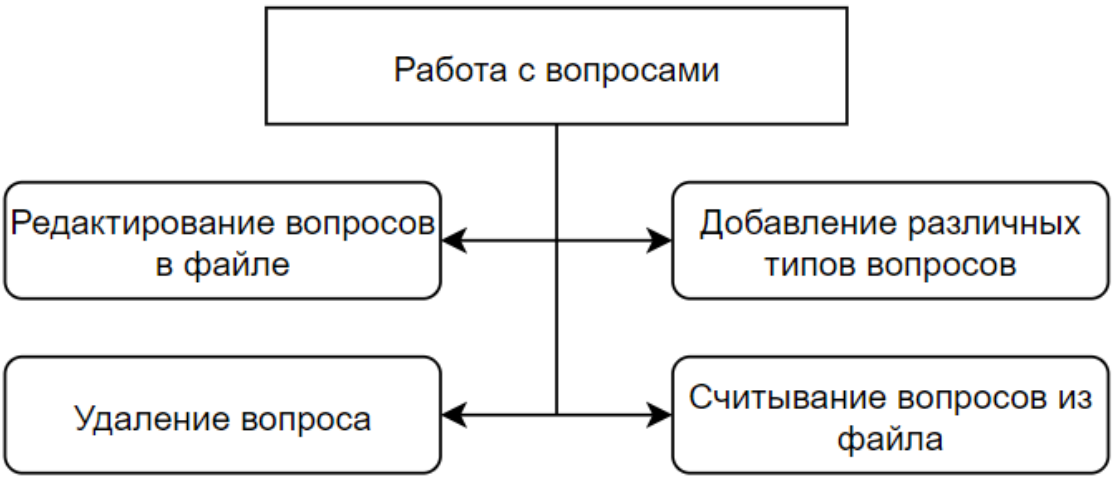

Рисунок 5 – Отображение функционала для работы с вопросами

Модуль Работа с вопросами разделяется на 4 подмодуля:

а) Редактирование вопросов в файле, где каждый тип вопроса определяется автоматически. В зависимости от типа вопроса открывается соответствующее диалоговое окно.

б) Добавление различных типов вопросов, где пользователь выбирает один из семи типов вопросов и добавляет его в файл. Расстановка специальных символов должна быть автоматической и зависеть от того, что выбрал пользователь.

в) Удаление вопроса, где пользователь выбирает вопрос, который хочет удалить.

г) Считывание вопрос из файла происходит автоматически при открытии нового файла в формате GIFT. Тип вопроса также определяется автоматически.

2.3 Диаграмма прецедентов

Исходя из вышеописанного функционала приложения можно создать следующую диаграмму прецедентов, которая будет отображать какие действия сможет выполнить пользователь, использующий приложение.

Рисунок 6 – Диаграмма прецедентов

2.4 Выбор интегрированной среды разработки

Интегрированная среда разработки (integrated development environment - IDE) - набор инструментов для разработки и отладки программ, имеющий общую интерактивную графическую оболочку, поддерживающую выполнение всех основных функций жизненного цикла разработки программы - набор и редактирование исходного текста (кода), компиляцию (сборку), исполнение, отладку, профилирование [11].

Выбор IDE особенно важен, так как она должна удовлетворять следующим потребностям: иметь возможность создавать кроссплатформенные приложения, поддерживать язык c++ и предлагать простую реализацию

графического интерфейса для создаваемых приложений. И всем этим требованиям соответствует IDE Qt Creator (Рисунок 7).

С помощью Qt Creator можно создать и запустить программное обеспечение, поддерживаемое большинством современных операционных систем путем простой компиляции программы, без необходимости изменять исходный код. В Qt Creator есть все необходимые для разработки основные классы, начиная от элементов графического интерфейса и заканчивая классами для работы с сетью, базами данных и XML [12]. Для личного использования Qt Creator бесплатна, она полностью подходит для написания редактора файлов и имеет хорошую документацию.

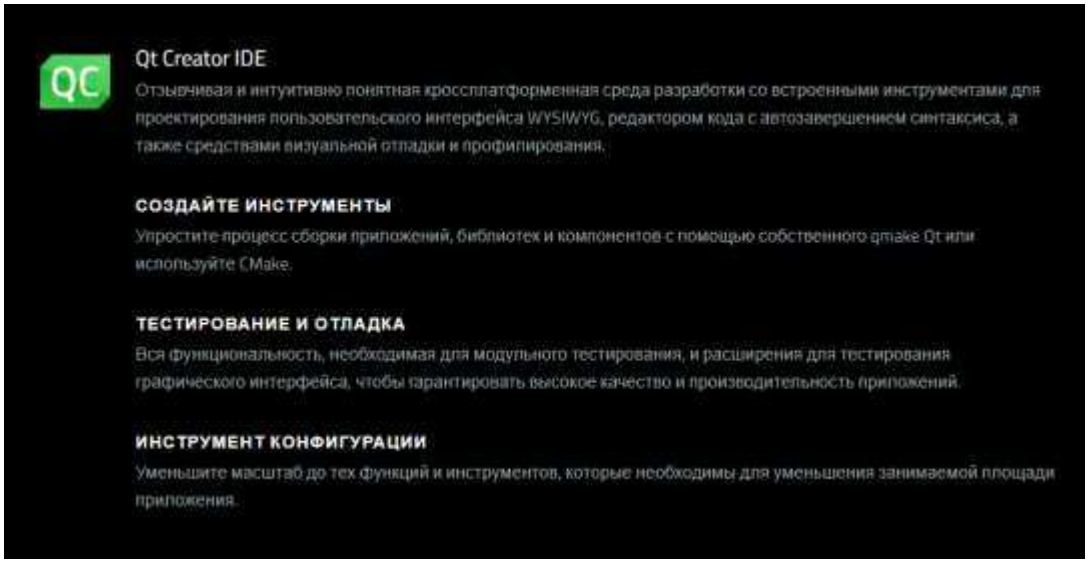

Рисунок 7 – Описание Qt Creator на официальном сайте компании Qt

2.5 Выбор архитектуры приложения

Для создания редактора файлов формата GIFT была выбрана архитектура Model-View-Controller (MVC).

Архитектура MVC подразумевает разделение кода приложения на 3 части: Модель (Model), Вид или Представление (View) и Контроллер (Controller). Такая архитектура впервые была описана в 1978 году, она использовалась для приложений, у которых был графический интерфейс. Чтобы упростить большой по объему код, необходимо разделить его на части.

Таким образом, в коде будет проще разобраться, внести необходимые изменения и минимизирует возможность допущения ошибки.

Более того, архитектура MVC не имеет привязки к конкретному языку программирования, не требует использования объектно-ориентированного программирования или какой-то другой парадигмы.

Разделение на части здесь не значит, что в проекте должно быть ровно 3 файла с названиями model, view и controller. На практике для каждого вида или класса делают отдельные папки.

Компоненты MVC приложения:

1. Модель. Основной задачей Модели является хранение и обработка данных, также она содержит в себе всю логику приложения. Модель не взаимодействует с пользователем напрямую, а обратиться к ней можно только из кода, вызывая ее функции. Например, Модель сохраняет информацию в БД, проверяет правильность введенных в форму данных, но при этом, она не может получать данные от пользователя или обрабатывать нажатие на кнопки, так как это не входит в ее задачи. Также Модель не должна зависеть от чего-либо, ей также неизвестно о Контроллерах и Видах;

2. Представление. Представление отображает данные, которые ему передали. В настольных или мобильных приложениях Представление — это код, который отвечает за отображение информации на экране и графический интерфейс;

3. Контроллер. Основными задачами Контроллера являются: выполнение запросов, поступивших от пользователя, а также обращение к модели, для получения или изменения данных, вызов Представления, для того, чтобы отобразить результат.

В настольных приложениях Контроллер отвечает за обработку нажатий на кнопки и других воздействий от пользователя. Один Контроллер может

работать с несколькими Моделями, и наоборот, одна Модель может использоваться в нескольких Контроллерах;

Взаимодействие между компонентами MVC может реализовываться разными способами в серверных и настольных приложениях. Для настольных приложений обычно выбирают «активную» модель, в то время как для серверных приложений чаще выбирают «пассивную». Этот выбор обусловлен тем, что Активная модель позволяет получать уведомления об изменениях в ней, а в серверных приложениях такая функция не требуется.

В случае с редактором файлов следует использовать активную модель, схема которой, изображает взаимодействие компонентов (Рисунок 8).

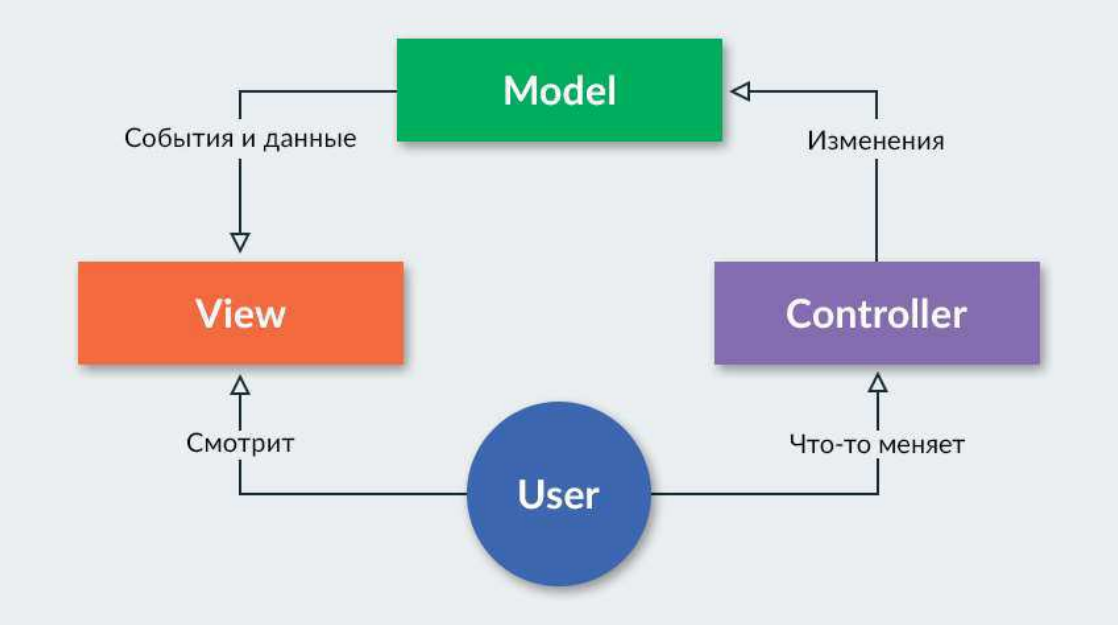

Рисунок 8 - Схема активной модели MVC

2.6 Структура приложения

При проектировании приложения первичной задачей является разбиение приложения на функциональные модули, необходимые для реализации. Для решения данной задачи можно использовать различные шаблоны проектирования. Для данного приложения была выбрана архитектура Model-

View-Controller, поэтому необходимо описать каждый модуль и то, как он будет применён к приложению.

2.6.1 Структура модели

В Модели приложения должна быть сущность, в которой будут сохраняться все вопросы, добавленные пользователей. Сущность, которая будет описывать различные типы вопросов, также должна находиться в Модели.

Для каждого типа вопросов необходимо сделать отдельный класс с уникальными полями. Сущности с типами вопросов будут наследоваться от одного абстрактного класса: суть данного подхода заключается в том, что у всех типов вопросов есть одинаковые параметры такие как заголовок и текст, но типы ответов разные.

Хранение вопросов можно реализовать с помощью контейнера.

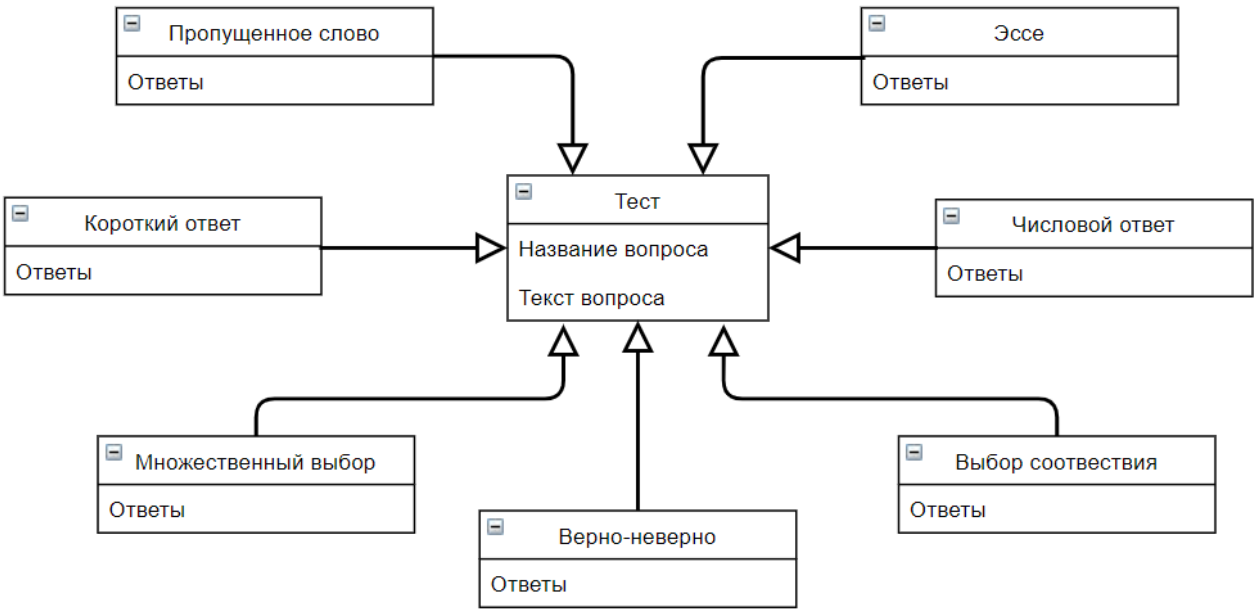

Рисунок 9 – Диаграмма сущностей вопросов

2.6.2 Структура контроллера

Контроллер приложения будет реагировать на следующие действия пользователя: добавление, удаление и редактирование вопросов, а запросы на выполнения данных действий будут посылаться в Модель.

При открытии нового файла Контроллер будет его обрабатывать. Для решения данной задачи следует создать отдельную сущность, которая будет считывать весь текст в файле, форматировать и отправлять в Модель, а данную сущность можно обозначить как «Парсер».

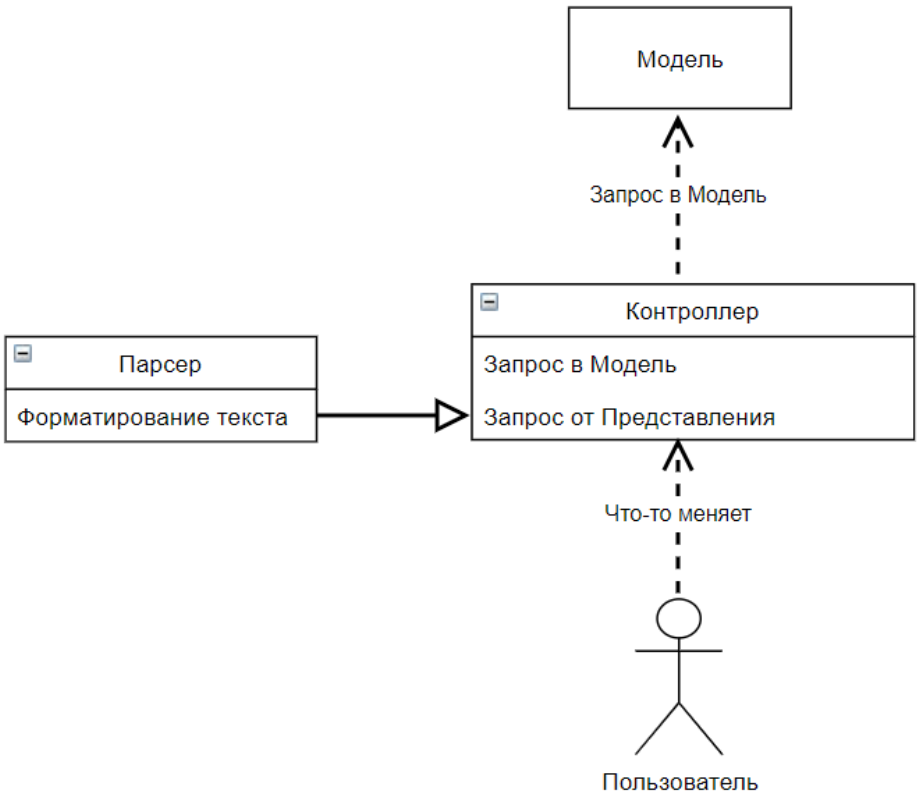

Рисунок 10 – Диаграмма Контроллера

2.6.3 Структура представления

Представление приложения должно отвечать за графический интерфейс пользователя, поэтому следует создать сущность, которая будет отвечать за главное диалоговое окно. В главном окне пользователь должен видеть все

функции приложения, а реализовать данную возможность можно с помощью кнопок.

Для каждого типа вопроса следует создать отдельное диалоговое окно, в котором пользователь сможет вписывать необходимые параметры. Должно быть 7 различных диалоговых окон и классов, взаимодействующих с друг с другом.

В ситуации, когда пользователь вносит какие-либо изменения в файл формата GIFT, Модель сообщает об этом Контроллеру, а после изменений, отображает их из Модели.

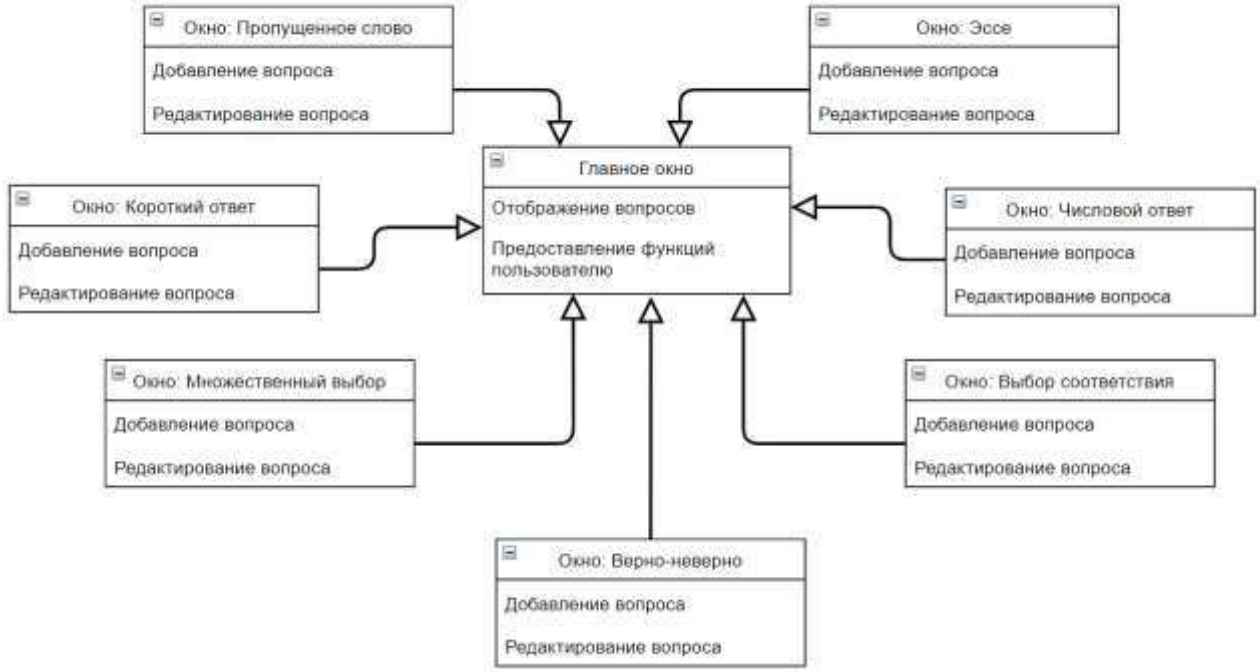

Рисунок 11 – Диаграмма диалоговых окон

3 Реализация приложения

Разработка редактора GIFT-файлов началась с создания Модели приложения, а Представление и Контроллер были созданы позднее. Такой подход использовался для того, чтобы Модель отвечала за все типы вопросов, а также за их хранение.

3.1 Реализация модели приложения

Был создан класс «ModelTest», в котором реализуются все необходимые методы для работы с вопросами. В данном классе была реализована абстрактная сущность Test, которая отвечает за все типы вопросов в файле, а от данной сущности наследуются конкретные классы для каждого типа вопроса.

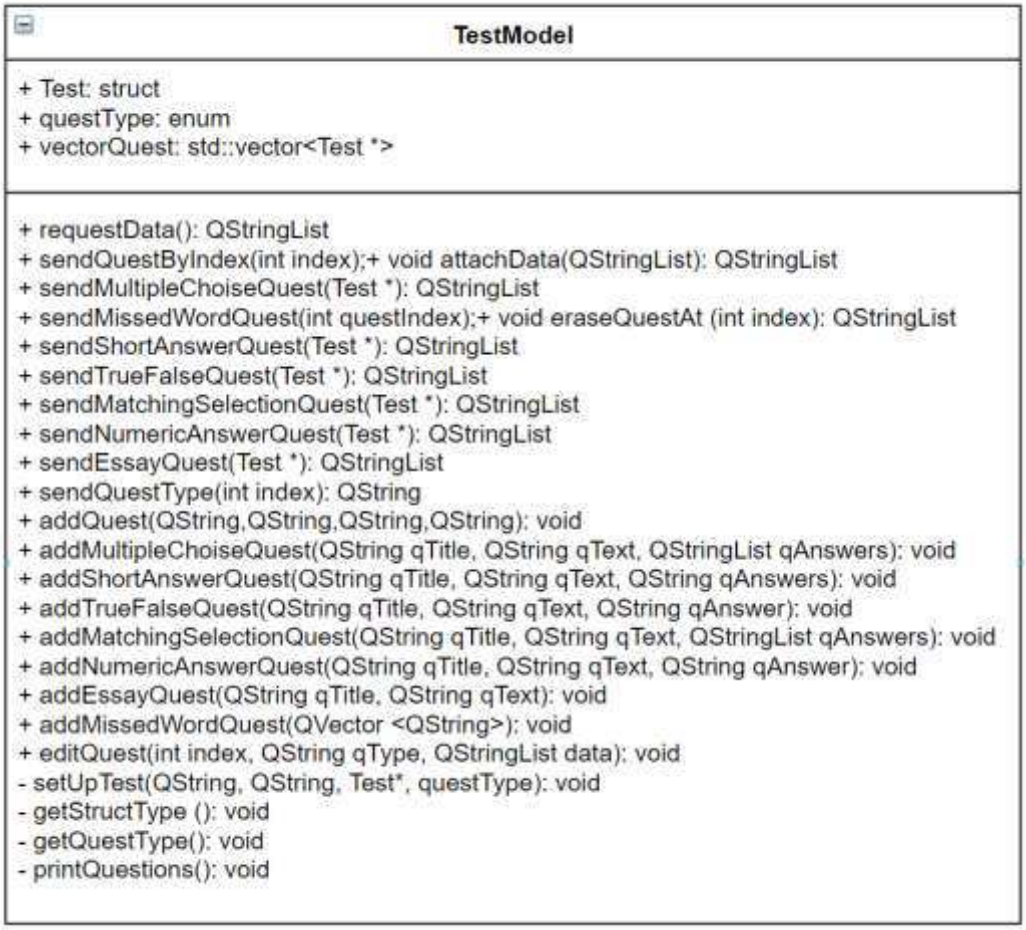

Рисунок 12 – Диаграмма класса «ModelTest»

Исходя из UML диаграммы сущностей вопросов (Рисунок 12), можно описать каждый класс, который отвечает за тот или иной тип вопроса:

«QuestMultipleChoice» - данная сущность реализует тип вопроса со множественным выбором. В переменной QStringList «answers» хранятся все варианты ответов;

«QuestTrueFalse» - реализует вопрос «Верно-неверно», а поле «answer» хранит ответ. Ответ записывается либо в полном формате (TRUE, FALSE), либо в сокращенном (T,F);

«QuestEssay» - реализует вопрос «Эссе» и в данной сущности нет полей, так как правильного ответа на данный тип вопроса не существует;

«QuestMissedWord» - сущность, созданная для вопроса «Пропущенное слово». Поля «firstHalf» и «secondHalf» хранят первую часть вопроса до вариантов ответа и вторую часть соответственно. Поле «answers» хранит ответы на вопрос;

«QuestNumericAnswer» - реализует вопрос с числовым ответом, а поле answer хранит ответ;

«QuestShortAnswer» - реализует вопрос с коротким ответом, поле answer хранит ответ;

«QuestMatchingSelection» - реализует вопрос «Выбор соответствия», а список «matchesArray» хранит соответствия между ответами;

Хранение вопросов было с реализовано с помощью контейнера «vectorQuest». Данный контейнер может содержать в себе элементы сущности Test.

Логика модели реализована с помощью методов класса «ModelTest». Основными функциями являются: удаление, добавление и редактирование вопросов, а данные функции взаимодействуют с контейнером, в котором хранятся все вопросы.

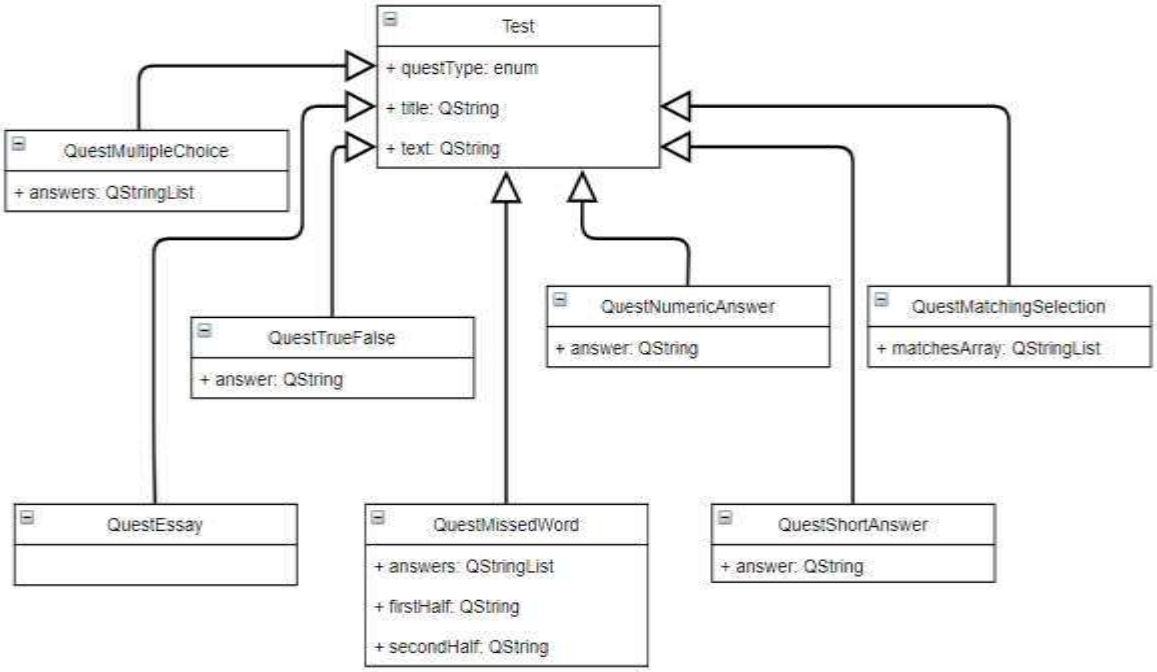

Рисунок 13 – UML диаграмма сущностей вопросов

3.2 Реализация представления приложение

Был создан класс «MainWindow», который является главным окном приложения. Для данного класса была разработана специальная форма, а главное окно отображается при запуске приложения.

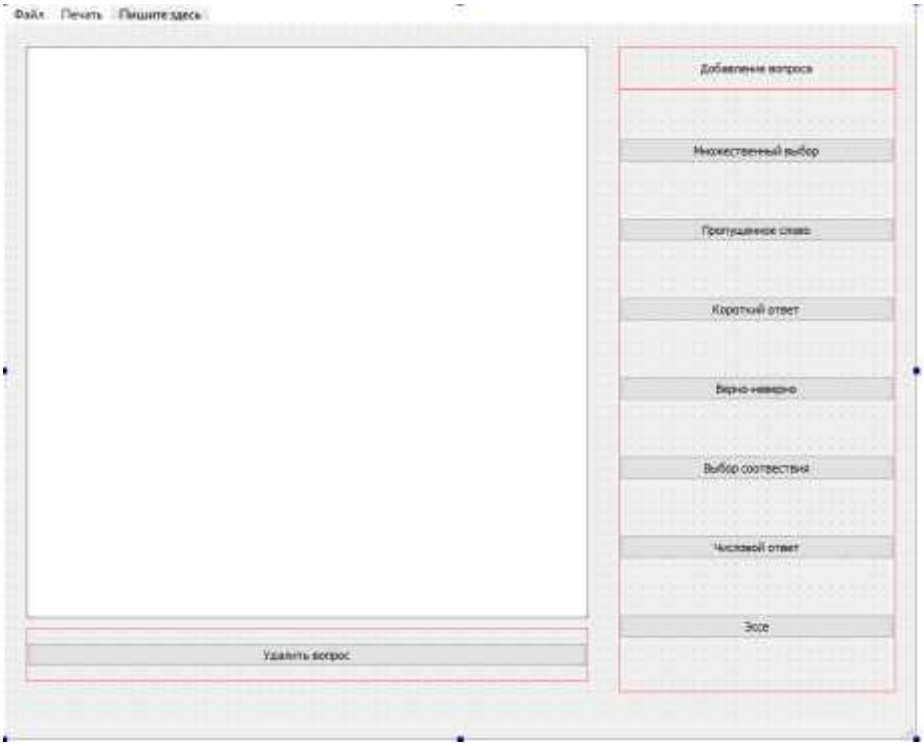

Рисунок 14 – Создание формы для главного окна приложения

Созданная форма «mainwindow.ui» (Рисунок 14) подключена к классу «MainWindow». Для отображения вопросов в GIFT файле используется QListWidget, где каждый элемент является каким-либо вопросом. Кнопки были подключены к различным функциям с помощью механизма сигналов и слотов в Qt. В menubar добавлены следующие дествия: открыть файл, сохранить файл и новый файл, а также печать текста из файла.

Для каждого типа вопроса необходимо создать уникальную форму пользовательского интерфейса и класс. С помощью инструментов Qt созданные формы были подключены к соответствующим классам, в которых была реализована логика работы (рисунок 15).

Классы, отвечающие за отображение и работу форм пользовательского интерфейса:

а) «DialogMatchingSelection» - отображает форму для вопроса на выбор соответствия;

б) «DialogMultipleChoice» - отображает форму для вопроса с множественным выбором;

в) «DialogEssay» - отображает форму для вопроса «Эссе»;

г) «DialogShortAnswer» - отображает форму для вопроса с коротким ответом;

д) «DialogTrueFalse» - отображает форму для вопроса «Верно-неверно»;

е) «DialogMissedWord» - отображает форму для вопроса с пропущенным словом;

ж) «DialogNumericAnswer» - отображает форму для вопроса с числовым ответом;

Рисунок 15 – Соответствие между формами и классами вопросов

При использовании приложения, пользователь работает только с Представлением, но для того чтобы Модель изменялась необходимо добавить Контроллер.

3.3 Реализация контроллера приложения

Контроллер должен получать сообщение от Представления о том, какое действие пользователь выбрал, а после передавать в Модель. Были созданы два класса: «ControllerApp» и «ControllerParserFile».

Класс «ControllerApp» взаимодействует с Моделью приложения. В нём были реализованы методы о желании пользователя внести изменения в контейнер вопросов.

Класс «ControllerParserFile» форматирует текст из выбранного пользователем файла, конвертируя его в формат UFT-8. Данный класс формирует список всех вопросов из файла и передаёт его в Модель, чтобы она добавила их в контейнер.

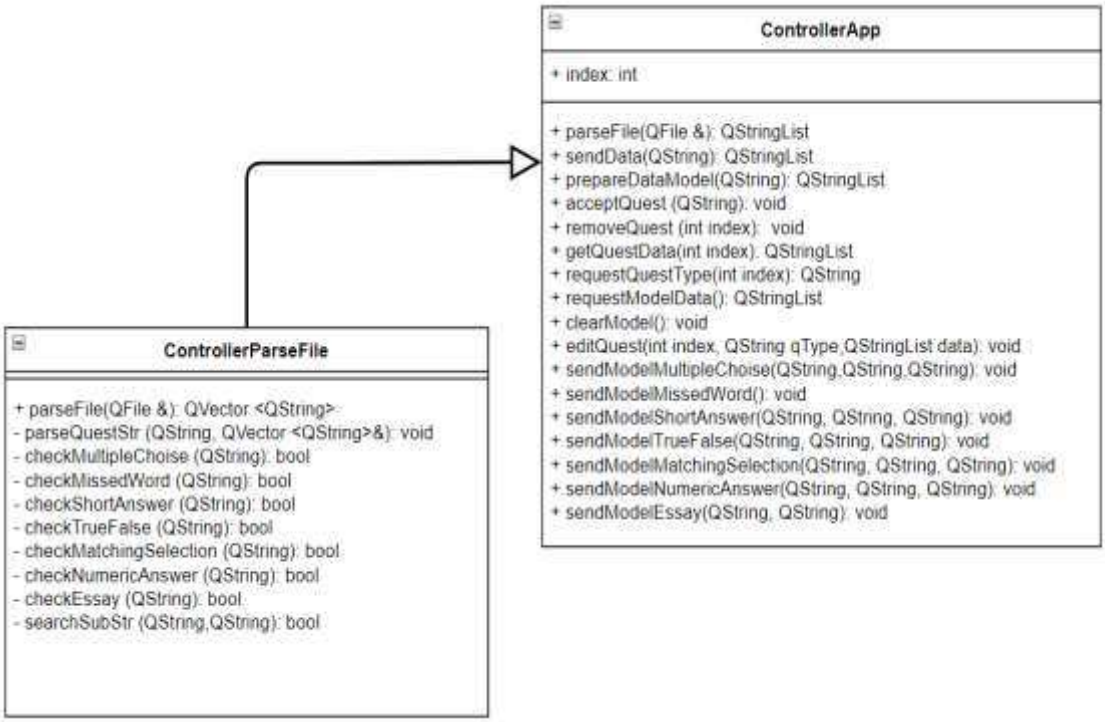

Рисунок 16 – UML диаграмма контроллера приложения

3.4 Разработка графического интерфейса

В приложении было создано 8 диалоговых окон, где каждое окно отвечает за определённое действие.

3.4.1 Главное окно

Главное окно было разделено на два блока (Рисунок 17). В правом блок создано 7 кнопок для добавления в файл нового вопроса. В зависимости от того, какой вопрос необходимо добавить в файл, пользователь выбирает соответствующую кнопку. В левом блоке представлен список всех вопросов в открытом файле формата GIFT и кнопка для удаления выбранных вопросов. В строке меню есть два списка: работа с файлами и печать.

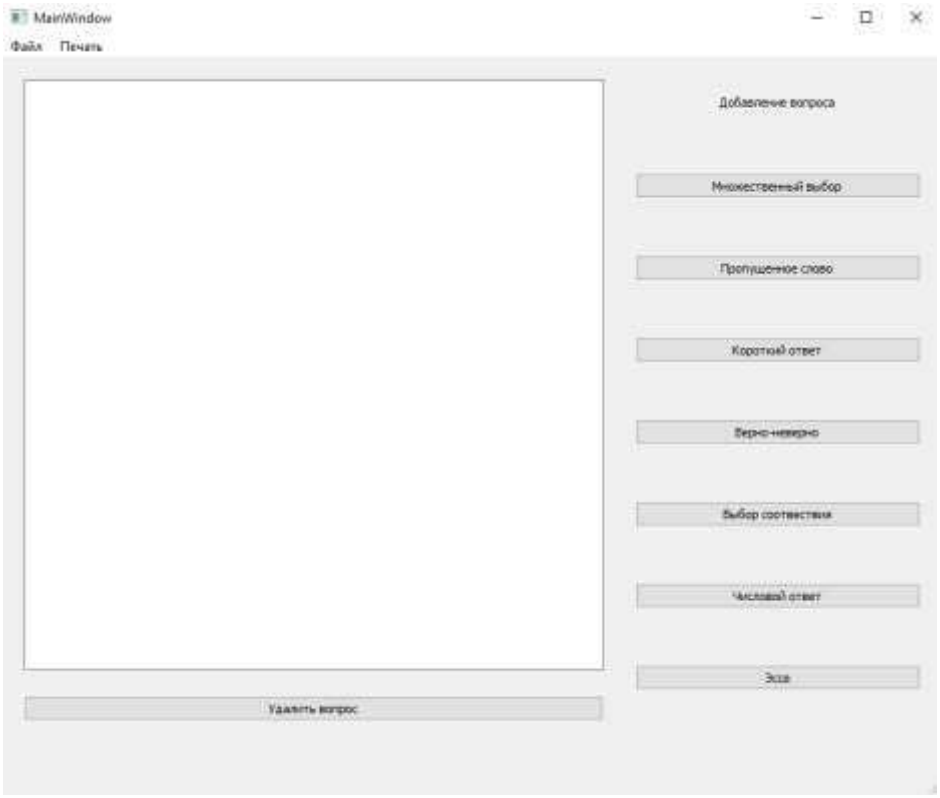

Рисунок 17 – Главное окно приложения

3.4.2 Окна для различных вопросов

Все окна для вопросов имеют примерно одинаковую конструкцию: для всех вопросов необходимо поле для ввода заголовков, текста и ответов, но так как ответы для каждого типа вопроса разные, соответственно поля для ответов тоже будут отличаться.

Окно для множественного выбора имеет две кнопки, два поля и блок со списком ответов (Рисунок 18).

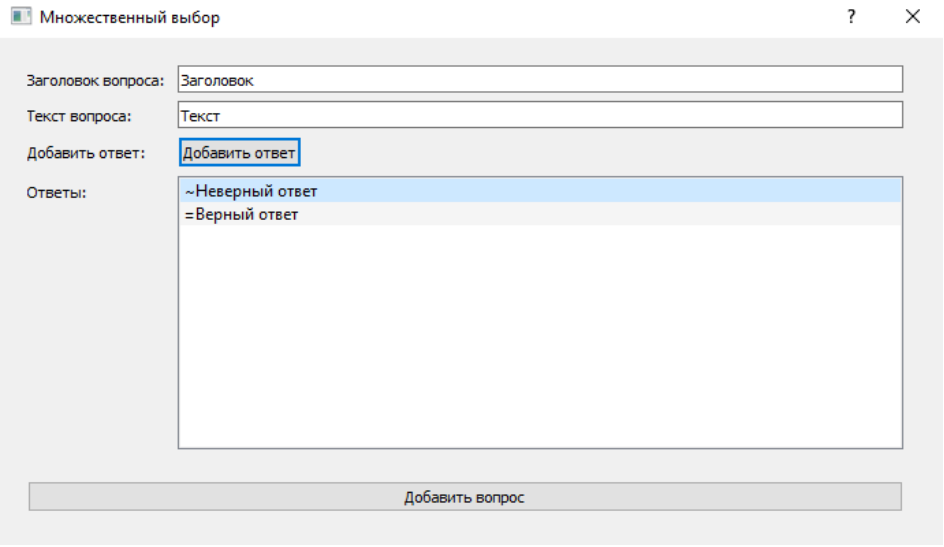

Рисунок 18 – Окно «Множественный выбор»

Окно для вопроса с пропущенным словом имеет 5 полей: заголовок вопроса, текст вопроса, первая часть предложения, вторая часть предложения и ответы (Рисунок 19).

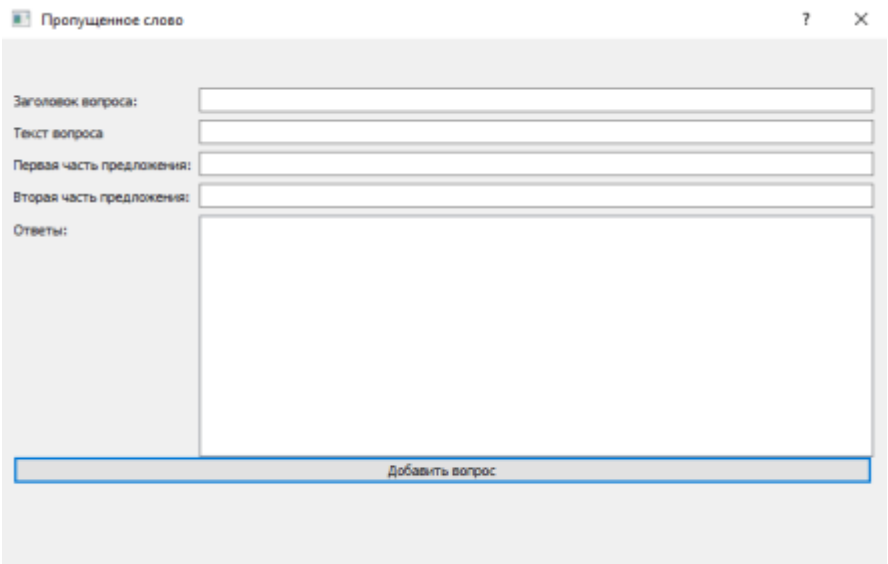

Рисунок 19 – Окно «Пропущенное слово»

Окно для вопроса с коротким ответом содержит три поля: заголовок вопроса, текст вопроса и ответ (Рисунок 20).

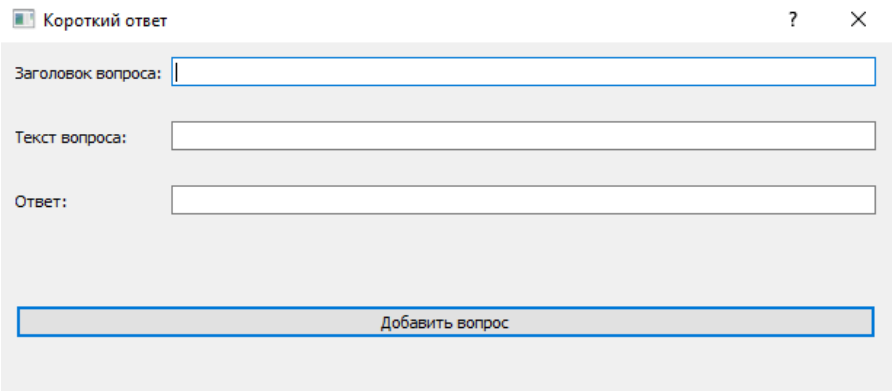

Рисунок 20 – Окно «Короткий ответ»

Окно для вопроса «Верно-неверно» имеет два поля: заголовок вопроса и текст вопроса, а также checkbox для выбора варианта верно или неверно (Рисунок 21).

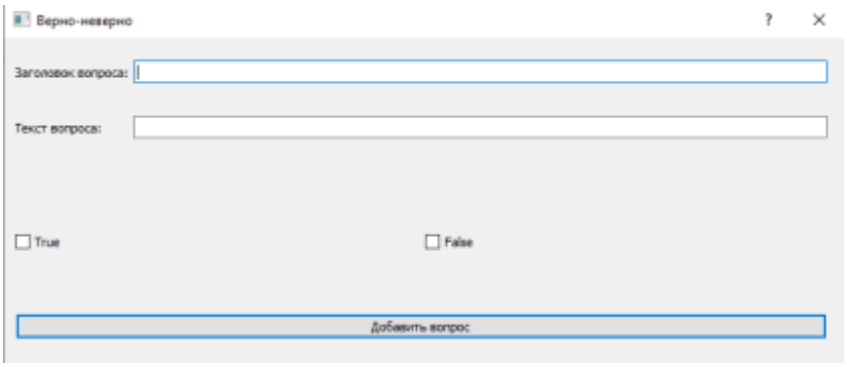

Рисунок 21 – Окно «Верно-неверно»

Окно для вопроса с выбором соответствия имеет два поля: заголовок вопроса и текст вопроса, а также таблицу, где пользователь выставляет соответствие между вариантами ответов (Рисунок 22).

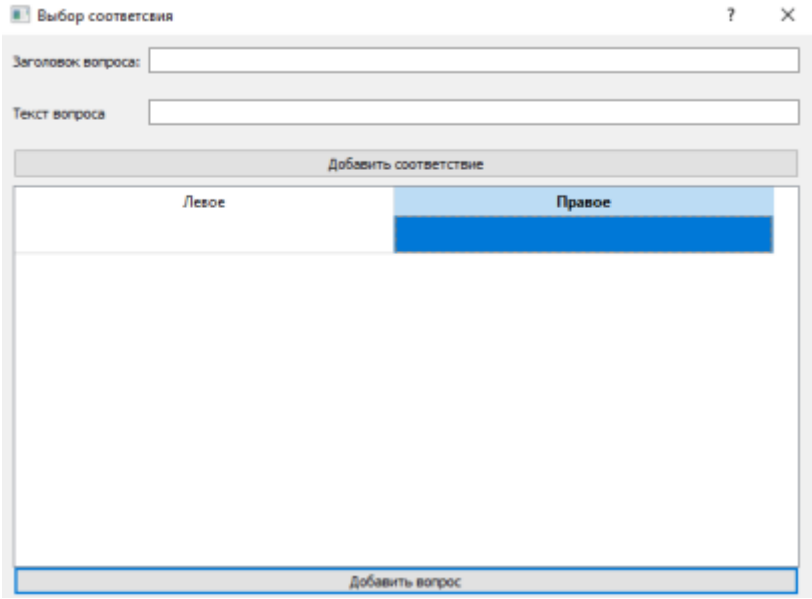

Рисунок 22 – Окно «Выбор соответствия»

Окно числовой ответ имеет три поля: заголовок вопроса, текст вопроса и ответ (Рисунок 23).

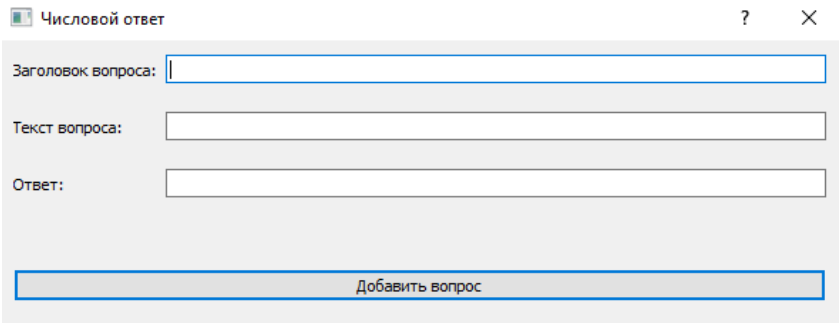

Рисунок 23 – Окно «Числовой ответ»

Окно вопроса «Эссе» имеет два поля: заголовок вопроса и текст вопроса (Рисунок 24).

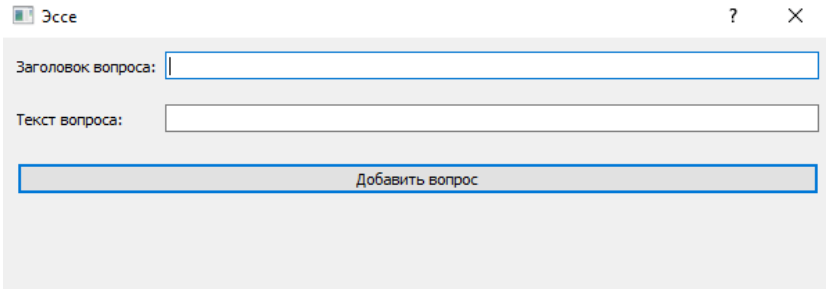

Рисунок 24 – Окно «Эссе»

3.5 Реализация работы с файлами

Одной из важнейших задач приложения является работа с файлами, поэтому были реализованы следующие функции: открытие, сохранение и создание файлов. Если пользователь создал или экспортировал GIFT файлы не в формате UTF-8, то приложение конвертирует входящий текст в правильную формат для системы Moodle. Для решения данной задачи был создан класс «ControllerParserFile», в одном из методов которого, при помощи библиотеки «QTextCodec», входящий текст конвертируется в формат UTF-8.

3.5.1 Открытие файлов

В главном окне пользователь может выбрать пункт «Открыть файл», после чего откроется диалоговое окно со всеми каталогами ОС (Рисунок 25), где можно выбрать нужный файл. Приложение может открывать файлы со следующими расширениями: «.gift», «.GIFT», «.txt».

После того, как пользователь выберет файл, класс «MainWindow» получит ссылку на этот файл и передаст её в класс «ControllerParserFile», а данный класс отформатирует текст в файле по переданной ссылке в формат UTF-8. Далее, текст будет разбит на определённое количество вопросов, исходя из количества пустых строку между ними. Найденные в тексте вопросы будут разделены по категориям и отправлены в класс «ModelTest», после чего добавлены в контейнер.

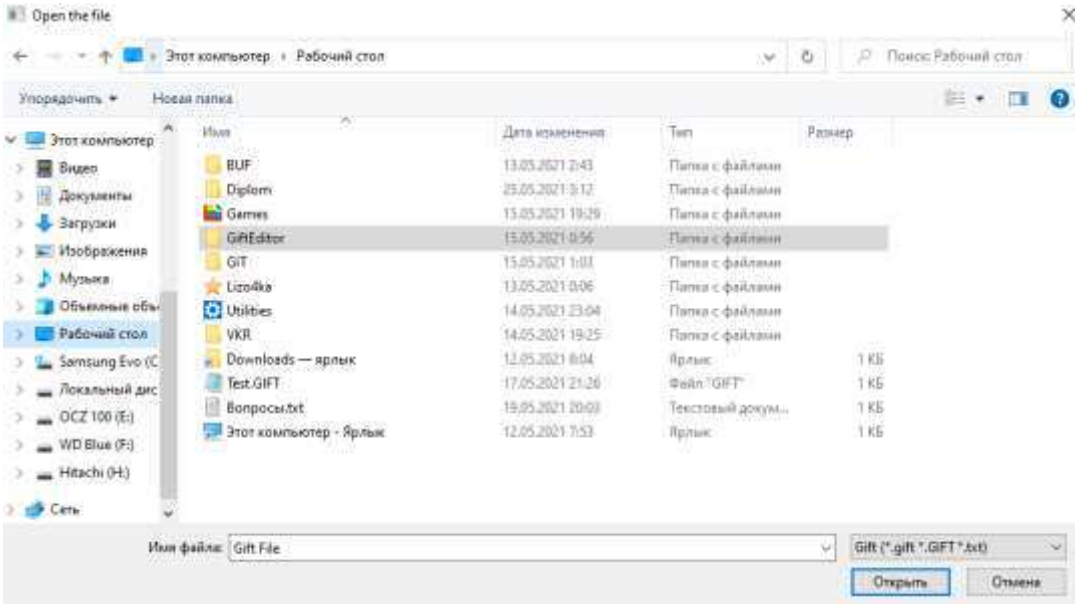

Рисунок 25 – Окно выбора файла

3.5.2 Сохранение файлов

Если пользователь выбрал пункт «Сохранить файл», то откроется диалоговое окно, в котором необходимо выбрать каталог для сохранения файла (Рисунок 26). Ссылку на каталог для сохранения файла получит класс «MainWindow» и запросит у класса «ControllerApp» актуальный список вопросов. «ControllerApp» обратится к классу «ModelTest», чтобы получить список вопросов в текстовом формате. В каталоге создастся новый файла и в него спишется полученный список вопросов.

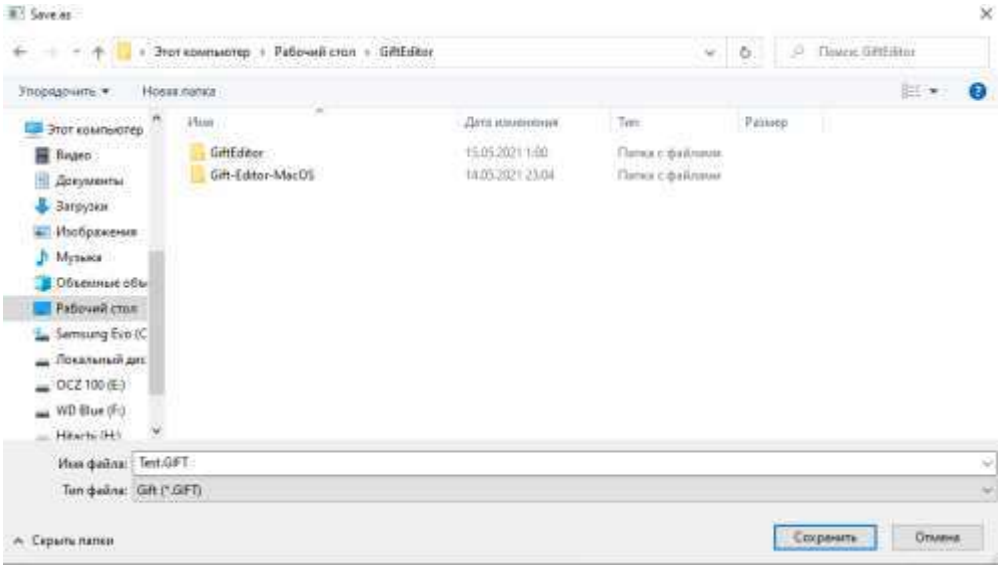

Рисунок 26 – Окно сохранения файла

3.5.3 Создание нового файла

Если пользователь выбрал пункт «Новый файл», то класс «MainWindow» сообщит классу «ControllerApp» о желании пользователя создать новый файл. «ControllerApp» вызовет метод в классе «ModelTest», который удалит все вопросы из контейнера вопросов. Так как список вопросов пуст, то класс «MainWindow» очистит QListWidget, в котором отображаются все вопросы.

3.6 Реализация работы с вопросами

Исходя из того, какую кнопку нажмёт пользователь, класс «MainWindow» откроет соответствующее диалоговое окно. После того, как пользователь введет все необходимые данные и нажмёт на кнопку «Добавить вопрос», их получит «ControllerApp», определит тип вопроса и выполнит запрос на добавление в контейнер вопросов класса «ModelTest». Так как список вопросов изменился, класс «MainWindow» должен обновить listWidget и для этого класс «ModelTest» передаёт актуальный список вопросов из контейнера.

В главном окне, при двойном нажатии левой кнопкой мыши на вопрос из списка, класс «MainWindow» запросит у класса «ControllerApp» информацию о том, какой тип вопроса был выбран. «ControllerApp» запросит информацию у класса «ModelTest», о том, какой тип вопроса хранится в контейнере вопросов под индексом выбранным пользователем, после чего «ModelTest» отправит всю информацию о вопросе в класс «MainWindow». Класс «MainWindow» инициализирует объект класса с соответствующим типом вопроса, передав в него всю информацию о вопросе, и откроет соответствующее диалоговое окно. Поля в открытом диалоговом окне будут заполнены данными из вопроса, который выбрал пользователь для редактирования. После того, как пользователь закончит редактирование вопроса и нажмёт на кнопку «Добавить вопрос», класс «MainWindow» передаст все данные, которые пользователь изменил в вопросе в «ControllerApp» и обновит отображаемый список вопросов из «ModelTest».

ЗАКЛЮЧЕНИЕ

В процессе выполнения выпускной квалификационной работы была достигнута поставленная цель – создание кроссплатформенного приложения для редактирования файлов формата GIFT. Для достижения данной цели были выполнены следующие задачи:

Был проведен анализ предметной области, благодаря которому определён список функциональных требований и получено представление о создаваемой системе;

Было проведено проектирование приложения, а именно выбрана архитектура Model-View-Controller. С учётом функциональных требований, была создана диаграмма прецедентов и выбрана интегрированная среда разработки Qt, что позволило не только реализовать необходимые задачи, но и сделать приложение кроссплатформенным.

Были разработаны различные диаграммы классов, что позволило реализовать кроссплатформенное приложение, которое полностью удовлетворяет поставленным требованиям. Созданный редактор файлов формата GIFT имеет возможность редактирования экспортированных файлов, например, из банка вопросов системы Moodle. Также реализованное приложение может создавать новые файлы и сохранять их в каталоге, выбранным пользователем. Были реализованы функции для добавления 7 типов вопросов: «множественный выбор», «пропущенное слово», «короткий ответ», «числовой ответ», «выбор соответствия», «эссе» и «верно-неверно». Созданное приложения позволяет печатать текст из редактируемого файла с помощью принтера или экспорт в PDF-документ.

Данная бакалаврская работа написана в соответствии с нормами СТО 4.2 -07-2014 Система менеджмента качества [13].

СПИСОК СОКРАЩЕНИЙ

ANSI - American National Standards Institute (Национальный институт стандартизации США).

IDE - Integrated Development Environment (Интегрированная среда разработки).

MVC - Model-View-Controller.

MacOS - Macintosh Operating System.

Moodle - Modular Object-Oriented Dynamic Learning Environment (модульная объектно-ориентированная динамическая обучающая среда).

UTF - Unicode Transformation Format (Формат Преобразования Юникод).

UML - Unified Modeling Language (Унифицированный язык моделирования).

XML - eXtensible Markup Language (расширяемый язык разметки).

ОС – операционная система.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Информатизация образования [Электронный ресурс]: Страница электронного обучения. URL: https://hotuser.ru/distanczionnoe-obuchenie/1142--e-learning (дата обращения 13.05.2020)

2. iSpring платформа для корпоративного обучения [Электронный ресурс]: Страница системы электронного обучения и тестирования Moodle: обзор возможностей. URL: https://www.ispring.ru/elearning-insights/moodle (дата обращения 13.05.2020)

3. MoodleDocs [Электронный ресурс]: Страница тестов в Moodle. URL: https://docs.moodle.org/310/en/Quiz_activity (дата обращения 13.05.2020)

4. iSpring платформа для корпоративного обучения [Электронный ресурс]: Страница обзора 6 платформ и сервисов для онлайн-обучения: возможности и решаемые бизнес-задач. URL: https://www.ispring.ru/elearning-insights/platforma-onlain-obucheniya (дата обращения 13.05.2020)

5. Хабр [Электронный ресурс]: Страница работы кодировки текста. URL: https://habr.com/ru/post/478636/ (дата обращения 15.05.2020)

6. MoodleDocs [Электронный ресурс]: Страница банка вопросов в Moodle. URL: https://docs.moodle.org/310/en/Question_bank (дата обращения 15.05.2020)

7. MoodleDocs [Электронный ресурс]: Страница импорта вопросов в Moodle. URL: https://docs.moodle.org/310/en/Import_questions (дата обращения 15.05.2020)

8. Национальная библиотека им. Н.Э. Баумана [Электронный ресурс]: Страница формата файлов. URL: https://ru.bmstu.wiki/%D0%A4%D0%BE%D1%80%D0%BC%D0%B0%D1%82_%D1%84%D0%B0%D0%B9%D0%BB%D0%BE%D0%B2 (дата обращения 17.05.2020)

9. MoodleDocs [Электронный ресурс]: Страница формата GIFT. URL: https://docs.moodle.org/311/en/GIFT_format#Format_symbols (дата обращения 20.05.2020)

10. Брянский государственный технический университет [Электронный ресурс]: Страница инструкции импорта тестов в LMS MOODLE. URL: https://www.tu-bryansk.ru/upload/medialibrary/2a4/%D0%98%D0%BD%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%86%D0%B8%D1%8F.%20%D0%98%D0%BC%D0%BF%D0%BE%D1%80%D1%82.pdf (дата обращения 21.05.2020)

11. Университет ИНТУИТ [Электронный ресурс]: Страница концепции современной интегрированной среды разработки приложений. URL: https://intuit.ru/studies/courses/13805/1223/lecture/23386?page=2 (дата обращения 21.05.2020)

12.Официальный сайт компании Qt [Электронный ресурс]: Страница продуктов и особенностей Qt. URL: https://www.qt.io/product/features (дата обращения 22.05.2020)

13. СТО 4.2-07-2014 Система менеджмента качества. Общие требования к построению, изложению и оформлению документов учебной деятельности. – Введ. 09.01.2014. – Красноярск: СФУ, 2014. – 60 с.

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт

Вычислительная техника
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой
_____ О.В. Непомнящий
подпись инициалы, фамилия
«10» 06 2021 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.01 Информатика и вычислительная техника
код и наименование направления

Редактор файлов формата GIFT
тема

Пояснительная записка

Руководитель _____ 08.06.21 доцент, рук. МУЛ ТП каф. ВТ Д.А. Швец
подпись, дата должность, ученая степень инициалы, фамилия

Выпускник _____ 09.06.21 Д.А. Коробкин
подпись, дата инициалы, фамилия

Нормоконтролер _____ 09.06.21 доцент, рук МУЛ ТП каф ВТ Д.А Швец
подпись, дата должность, ученая степень инициалы, фамилия

Красноярск 2021