Министерство науки и высшего образования Российской Федерации

федеральное государственное бюджетное образовательное учреждение

высшего образования

«Российский экономический университет имени Г.В. Плеханова»

Московский приборостроительный техникум

Выпускная квалификационная работа

(Дипломная работа)

|  |  |
| --- | --- |
| На тему: | Разработка системы управлением тестирование и мониторинга виртуальной среды (на примере ООО «Фаматек») |

АЛЕКСЕЕВА СЕМЁНА АЛЕКСАНДРОВИЧА

Студент (-ка) 4 курса группы Т50-1-19

по специальности 09.02.07 «Информационные системы и программирование»

для присвоения квалификации: специалист по тестированию в области информационных технологий

Форма обучения: очная

Руководитель: / Комаров Андрей Алексеевич /

(подпись)

« » 2023 г.

Консультант: / Варламова Дарья Сергеевна /

(подпись)

« » 2023 г.

Студент (-ка): / Алексеев Семён Александрович /

(подпись)

« » 2023 г.

Допущена к защите

Приказ от «18» мая 2023 г. № 18.01-354

2023

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 2](#_Toc131771636)

[1. ОБЩАЯ ЧАСТЬ 2](#_Toc131771637)

[1.1. Цель разработки 2](#_Toc131771638)

[1.2. Средства разработки 2](#_Toc131771639)

[2. СПЕЦИАЛЬНАЯ ЧАТЬ 2](#_Toc131771640)

[2.1. Постановка задачи 2](#_Toc131771641)

[2.1.1. Требования к проекту 2](#_Toc131771642)

[2.2. Внешняя спецификация 2](#_Toc131771643)

[2.2.1. Описание задачи 2](#_Toc131771644)

[2.2.2. Входные и выходные данные 2](#_Toc131771645)

[2.2.3. Методы 2](#_Toc131771646)

[2.2.4. Тесты 2](#_Toc131771647)

[2.3. Проектирование 2](#_Toc131771648)

[2.3.1. Инфологическая модель 2](#_Toc131771649)

[2.3.2. Даталогическая модель 2](#_Toc131771650)

[2.3.3. Словарь данных 2](#_Toc131771651)

[2.3.4. Структурная схема 2](#_Toc131771652)

[2.3.5. Функциональная схема 2](#_Toc131771653)

[2.3.6. Диаграмма классов 2](#_Toc131771654)

[2.3.7. Схема тестирования 2](#_Toc131771655)

[2.3.8. Схема пользовательского интерфейса 2](#_Toc131771656)

[2.3.9. Результат работы программы 2](#_Toc131771657)

[3. ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ 2](#_Toc131771658)

[3.1. Инструментальные средства 2](#_Toc131771659)

[3.2. Отладка программы 2](#_Toc131771660)

[3.3. Защитное программирование 2](#_Toc131771661)

[3.4. Характеристика программы 2](#_Toc131771662)

[ЗАКЛЮЧЕНИЕ 2](#_Toc131771663)

[ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ 2](#_Toc131771664)

ПРИЛОЖЕНИЕ А. ТЕКСТ ПРОГРАММЫ

ПРИЛОЖЕНИЕ Б. СЦЕНАРИЙ И РЕЗУЛЬТАТЫ ТЕСТОВЫХ ИСПЫТАНИЙ

ПРИЛОЖЕНИЕ В. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

ПРИЛОЖЕНИЕ Г. СКРИПТ БАЗЫ ДАННЫХ

## ВВЕДЕНИЕ

В современном мире разработка программного обеспечения является неотъемлемой частью бизнеса и технологического прогресса. Одним из ключевых этапов в разработке ПО является тестирование, которое позволяет выявить ошибки и дефекты программы до ее внедрения в рабочую версию продукта, с которой будет работать пользователь. Тестирование может занять продолжительное время и быть ресурсозатратным процессом, если необходимо проверить несколько приложений с разными конфигурациями персональных компьютеров.

В этом контексте разработка системы управления тестированием и мониторинга виртуальной среды становится важным заданием для IT-специалистов. Такое приложение может быть особенно полезным для команд отдела тестирования программного обеспечения и позволит им значительно сократить время и ресурсы, затрачиваемые на тестирование продуктов компании.

Достоинства такого приложения:

* Приложение позволяет значительно сократить время, затрачиваемое на тестирование. Автоматизация тестов позволяет запускать тесты одновременно на нескольких виртуальных машинах, что ускоряет процесс и сокращает количество времени, которое необходимо для тестирования;
* Позволяет существенно сократить затраты на тестирование, так как оно позволяет максимально эффективно использовать имеющиеся ресурсы и не требует дополнительных затрат на приобретение нового оборудования, так как виртуальные машины можно по-разному настраивать;
* Мониторинг результатов тестирования. Благодаря этому, тестировщики могут быстро выявлять ошибки и дефекты для повышения качества продукции;
* Обеспечивает более высокую точность тестирования, чем ручное тестирование. Автоматизированные тесты выполняются одинаково точно каждый раз, что повышает качество тестирования и уменьшает вероятность ошибок, связанных с человеческим фактором.

При выполнении дипломного проекта стоит задача разработать приложение для запуска автоматизированных тестов на удаленных виртуальных машинах или на локальном компьютере. Также в данном приложение должен иметься функционал сохранения настроек подключения к виртуальным машинам, последовательности тестов, как локально, так и в базу данных, файл журнала и результаты тестов в табличном виде с возможностью просмотра графика.

# ОБЩАЯ ЧАСТЬ

## Цель разработки

Повышение эффективности отдела тестирования и качества выпускаемых программных продуктов компанией, и ускорение процесса разработки.

## Средства разработки

В ходе разработки использовались технические средства, представленные в таблице 1.

Таблица – Технические средства

|  |  |  |
| --- | --- | --- |
| № | Тип оборудование | Наименование оборудование |
| 1 | 2 | 3 |
| 1 | Размер экрана: | 15,6" |
| 2 | Разрешение экрана: | 1920x1080 |
| 3 | Линейка процессора: | Intel Core i5 8300H |
| 4 | Количество ядер процессора: | 4 |
| 5 | Оперативная память: | 16 ГБ |
| 6 | Видеокарта: | GeForce GTX 1060 |
| 7 | Конфигурация накопителей: | SSD + HDD |
| 8 | Общий объем всех накопителей: | 256гб x2 + 1TB |
| 9 | Операционная система | Windows 11 Pro |

В таблице 2 представлены минимальные и рекомендованные технические средства технические средства для комфортного использования программы.

Таблица – Минимальные и рекомендованные технические средства

| № | Тип оборудование | Наименование оборудование |
| --- | --- | --- |
| 1 | 2 | 3 |
| Минимальные характеристики | | |
| 1 | Размер экрана: | 13" |
| 2 | Разрешение экрана: | 1280x1024 |
| 3 | Линейка процессора: | Intel Pentium G6400 |
| 4 | Количество ядер процессора: | 2 |
| 5 | Оперативная память: | 4 ГБ |
| 6 | Видеокарта: | UHD-graphics 610 |
| 7 | Конфигурация накопителей: | SSD |
| 8 | Общий объем всех накопителей: | 256Гб |
| 9 | Операционная система | Windows 10 Home |
| Рекомендуемые характеристики | | |
| 1 | Размер экрана: | 27" |
| 2 | Разрешение экрана: | 2560x1440 |
| 3 | Линейка процессора: | AMD Ryzen R5 2600 |
| 4 | Количество ядер процессора: | 6 |
| 5 | Оперативная память: | 32 ГБ |
| 6 | Видеокарта: | GeForce RTX 2060 |
| 7 | Конфигурация накопителей: | SSD |
| 8 | Общий объем всех накопителей: | 1Tб |
| 9 | Операционная система | Windows 11 Pro |

Состав программных средств, используемых при разработке информационной системы представлены в таблице 3.

Таблица – Программные средства

| № | Тип средства | Название средства | Назначение |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 1 | Операционная система | Microsoft Windows 11 22H2 | Организация взаимодействия программ и  пользователя |
| 2 | Инструментальное средство разработки программных решений | IntelliJ IDEA 2022.2.1 | Разработка программного интерфейса приложения |
| 3 | Инструментальное средство разработки программных решений | Microsoft Visual Studio Community 2022 17.3.3 | Разработка клиентского приложения |
| 4 | Система управлением версий | GitHub | Управление версиями приложения |
| 5 | Система управления базами данных | pgAdmin 4 | Разработка базы данных |
| 6 | Система автоматизации развертывания с поддержкой контейнеризации | Docker engine 23 | Развертывание программного интерфейса на сервере |

# СПЕЦИАЛЬНАЯ ЧАТЬ

## Постановка задачи

Необходимо разработать приложение для сбора файлов журнала, запуска автоматизированных тестов на виртуальных машинах или на локальном компьютере, с сохранением результатов, файлов настроек, на языке программирования C# с системой построения клиентского приложения WPF. Также необходимо разработать программный интерфейс для взаимодействия программы с базой данных на языке программирования Java с фрейворком Spring. Также, программный интерфейс должен быть запущен на виртуальном выделенном сервере с помощью системы контейнеризации Docker.

Информационная система разрабатывается для упрощения сбора файлов журнала, запуска виртуальных машин и запуска автоматизированных тестов.

## Требования к проекту

Разработать информационную систему сбора файлов журнала, запуска автоматизированных тестов на виртуальных машинах или на локальном компьютере, с сохранением результатов и файлов настроек. Также требуется реализовать функции добавления, изменения, удаления, экспорта данных, просмотра графика и скачивания файлов.

Важными требованиями к системе являются:

* Разграничение прав доступа пользователей;
* Защита данных с помощью шифрования и хеширования;
* Пользовательский интерфейс с дружественным использованием, открывающийся на весь экран и с модальными окнами;
* Функционал добавления, редактирования, удаления, экспорта данных и скачивания файлов;
* Инсталляционный проект для удобной установки приложения.

Приложение должно содержать следующие экраны:

1. Экран авторизации с полями «Email» и «Пароль», а также кнопками «Войти» и «Зарегистрироваться»;
2. Экран регистрации с полями «Фамилия», «Имя», «Отчество», «Email», «Пароль» и кнопками «Зарегистрироваться» и «Назад»;
3. Экран администратора с полями «Фамилия», «Имя», «Отчество», «Email», выпадающим списком ролей и таблицей пользователей, у поле «Фамилия», «Имя», «Отчество», «Email» и «Роль», а также кнопками «Изменить», «Удалить», «Обновить таблицу» и «Выход». На вкладке «Результаты» должно располагаться поле «Email» и кнопка «Загрузить»;
4. Главное окно, которое открывается после регистрации или авторизации, содержит несколько вкладок, такие как «Основной», «Локальный» «Настройки» и «Справка».

Во вкладке «Основной» должны располагаться следующие элементы: таблица «Виртуальные машины», состоящая из полей «Название виртуальной машины» отображаются все виртуальные машины из VirtualBox, в поле «Снапшоты» выпадающий список всех снимков данных для определенной виртуальной машины из VirtualBox, поле «off/on» с флаговой кнопкой и текстовое поле «PsExec line» . и «Тесты» состоящая из полей «Название теста», которое определяется из названия файла, поле «off/on» с флаговой кнопкой, неизменяемое тестовое поле «Путь» и текстовое поле «Параметр», кнопки «Добавить запись», «Удалить запись» и «Запустить», радиокнопка «Запустить в режиме GUI?», меню «Файл», в котором кнопки «Сохранить PsExec», «Открыть PsExec», «Сохранить порядок тестов» и «Загрузить порядок тестов», и меню «VirtualBox», в котором кнопки «Включить», «Поставить на паузу», «Продолжить», «Выключить», «Загрузить репозиторий», «Включить синхронизацию времени» и «Выключить синхронизацию времени».

Во вкладке «Локальный» располагаются следующие элементы: таблица «Тесты» состоящая из полей «Название теста», которое определяется из названия файла, поле «off/on» с флаговой кнопкой, неизменяемое тестовое поле «Путь» и текстовое поле «Параметр», кнопки «Добавить запись», «Удалить запись» и «Запустить», меню «Файл» с кнопками «Сохранить порядок тестов» и «Загрузить порядок тестов», меню «Настройки» с кнопкой «Загрузить репозиторий».

Во вкладке «Настройки» располагаются вкладки «Результаты тестов», «Виртуальные машины» и «Тесты». Во вкладке «Результаты тестов» вкладки «Результаты» и «График». Во вкладке «Результаты» располагается таблица со следующими полями: «Название теста», «Параметр», «Код выхода», «Дата» и «Результат». Во вкладке «График» располагается гистограмма и кнопка «Сохранить фото». Во вкладке «Виртуальные машины» находится таблица, у которой только поле «Название файла» и кнопки «Скачать», «Добавить» и «Удалить». Во вкладке «Тесты» находится таблица, у которой только поле «Название файла» и кнопки «Скачать», «Добавить» и «Удалить».

Во вкладке «Справка» располагаются следующие элементы: Логотип приложения, версия приложения, версия операционной системы, путь к исполняемому файлу и авторские права содержащий следующий текст «© AlexeevSA 2023г. Все права защищены.», кнопка «Проверить наличие обновления».

Описание функций после взаимодействия с кнопками программы:

После нажатия кнопки «Войти» происходит запрос в программный интерфейс с проверкой введенных данных, если данные верны, то генерируется токен и открывается главное окно пользователя или окно администратора в зависимости от роли, в ином случае выдается ошибка «Неверный логин или пароль».

После нажатия кнопки «Регистрация» происходит запрос в программный интерфейс с проверкой введенных данных, если таких данных еще нет в базе данных и они проходят все условия, то создается аккаунт и генерируется токен, после открывается главное окно пользователя, в ином случае выдается ошибка указывающая на неправильность действий.

После нажатия кнопки «Запустить» проверяется флаговая кнопка «Использовать виртуальную машину», если флажок поставлен, то проверяется значение из выпадающего списка, затем проверяется переключатель «Запустить в режиме гуи?», если значение стоит «Да», то выбранные виртуальные машины запускаются режиме графической оболочки, в ином случае без.

После нажатия кнопки «Пауза» проверяется флаговая кнопка «Использовать виртуальную машину», если флажок поставлен, то выбранные виртуальные машины переходят в состояние паузы.

После нажатия кнопки «Продолжить» проверяется флаговая кнопка «Использовать виртуальную машину», если флажок поставлен, то выбранные виртуальные машины выходят из состояния паузы.

После нажатия кнопки «PowerOff» проверяется флаговая кнопка «Использовать виртуальную машину», если флажок поставлен, то выбранные виртуальные машины выключаются.

После нажатия кнопки «Sync host on» проверяется флаговая кнопка «Использовать виртуальную машину», если флажок поставлен, то у выбранных виртуальных машин включается синхронизация со временем компьютера, с которого запущена виртуальная машина.

После нажатия кнопки «Sync host off» проверяется флаговая кнопка «Использовать виртуальную машину», если флажок поставлен, то у выбранных виртуальных машин отключается синхронизация со временем компьютера, с которого запущена виртуальная машина.

После нажатия кнопки «Загрузить репозиторий» проверяется флаговая кнопка «Использовать виртуальную машину», если флажок поставлен, то у выбранных виртуальных машин обновляется репозиторий.

После нажатия кнопки «Сохранить PsExec» открывается окно с выбором директории, куда необходимо сохранить файл, после введения названия и нажатия кнопки «Сохранить» файл с настройками PsExec line сохраняется.

После нажатия «Загрузить PsExec» открывается окно с выбором файла, который необходимо загрузить, после выбора и нажатия кнопки «Открыть» настройки PsExec line загружаются в таблицу.

После нажатия кнопки «Сохранить порядок тестов» открывается окно с выбором директории, куда необходимо сохранить файл, после введения названия и нажатия кнопки «Сохранить» файл с настройками тестов сохраняется.

После нажатия кнопки «Загрузить порядок тестов» открывается окно с выбором файла, который необходимо загрузить, после выбора и нажатия кнопки «Открыть» настройки тестов загружаются в таблицу.

После нажатия кнопки «Добавить запись» открывается окно с выбором файла, который необходимо добавить в таблицу тестов, после выбора нажатия кнопки «Открыть» файл добавляется в таблицу.

После нажатия кнопки «Удалить запись» открывается диалоговое окно с подтверждением удаления файла из таблицы, если пользователь нажмет кнопку «Да», то файл удалиться из таблицы, в ином случае окно закроется.

После нажатия кнопки «Изменить» на окне администратора проверяется выбран ли элемент в таблице и проверяются введены ли данные в тестовые поля, если все корректно, то происходит запрос в программный интерфейс и данные о пользователе изменяются, в ином случае выдается ошибка.

После нажатия кнопки «Удалить» на окне администратора проверяется выбран ли элемент в таблице, если всё корректно, то происходит запрос в программный интерфейс и пользователь удаляется. Пользователь не может удалить самого себя.

После нажатия кнопки «Обновить таблицу» на окне администратора происходит перерисовка и обновление таблицы.

После нажатия кнопки «Загрузить» на окне администратора проверяется введены ли данные в текстовое поле «Email», если всё корректно, то загружаются результаты тестирования веденного пользователя.

После нажатия кнопки «Сохранить фото» пользователю открывается окно с выбором директории, в которую будет сохранён график результатов тестирования.

После нажатия кнопки «Скачать» проверяется выбран ли элемент в таблице, если всё корректно, то скачивается выбранный файл, в ином случае выдается ошибка.

После нажатия кнопки «Добавить» открывается окно с выбором файла, если нажата кнопка «Сохранить», то происходит запрос в программный интерфейс и файл сохраняется в базе данных и обновляется таблица.

После нажатия кнопки «Удалить» проверяется выбран ли элемент в таблице, если всё корректно, то выбранный файл удаляется из базы данных и таблица обновляется.

После нажатия кнопки «Проверить наличие обновления» происходит запрос на сервер GitHub, на котором располагается исполняемый файл с указанием версии, если версия программы на сервере выше, то пользователю открывается модальное окно, в котором предлагается обновить программу, если версии одинаковы, то пользователю выдается сообщение, что у него самая актуальная версия.

## Внешняя спецификация

## Описание задачи

На рисунке 1 представлена диаграмма прецедентов, в которая отображает возможности пользователя в информационной системе, также представлены все возможные роли пользователя.

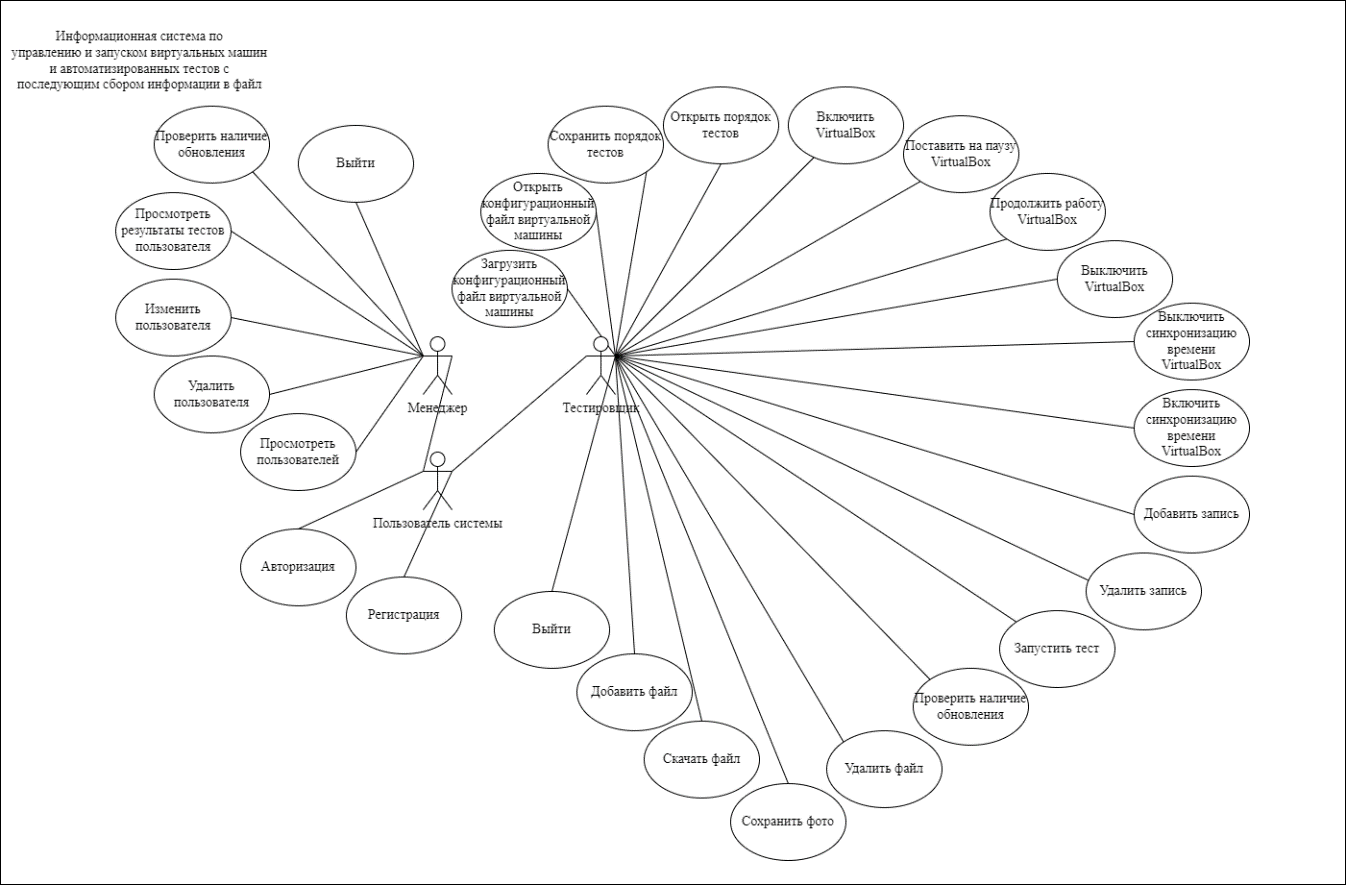


Рисунок – Диаграмма прецедентов

## Входные и выходные данные

В таблице 4 представлены входные данные, вводимые пользователем.

Таблица – Входные данные

| Имя | Ограничение | Тип данных | Описание |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 5 |
| Экран авторизации | | | |
| Email | ^\\S+@\\S+\\.\\S+$ | String | Строка электронной почты |
| Пароль | Не должно быть пустым и длиннее 8 символов | String | Строка пароля |
| Экран регистрации | | | |
| Фамилия | Не должно быть пустым | String | Строка с фамилией пользователя |
| Имя | Не должно быть пустым | String | Строка с именем пользователя |
| Отчество | Ограничений нет | String | Строка с отчеством пользователя |
| Email | ^\\S+@\\S+\\.\\S+$ | String | Строка электронной почты |
| Пароль | Не должно быть пустым и длиннее 8 символов | String | Строка пароля |
| Экран Администратора | | | |
| Фамилия | Не должно быть пустым | String | Строка с фамилией пользователя |
| Имя | Не должно быть пустым | String | Строка с именем пользователя |
| Отчество | Ограничений нет | String | Строка с отчеством пользователя |
| Email | ^\\S+@\\S+\\.\\S+$ | String | Строка электронной почты |
| Экран пользователя | | | |
| PsExec line | Не должно быть пустым | String | Строка удаленного подключения к виртуальной машине |
| Off/on | Ограничений нет | Checkbox | Флаговая кнопка включения виртуальной машины |
| Снапшот | Ограничений нет | Combobox | Выпадающий список со снимками состояния виртуальной машины |
| Параметр | Ограничений нет | String | Строка параметра теста |
| Запуск в режиме GUI | Ограничений нет | RadioButton | Переключатель режима запуска виртуальных машин |
| Конфигурационный файл виртуальной машины | Ограничений нет | Xml File | Файл с конфигурацией виртуальных машин |
| Конфигурационный файл тестов | Ограничений нет | File | Файл с конфигурацией тестов |

В таблице 5 представлены выходные данные информационной системы «по управлению и запуску виртуальных машин и автоматизированных тестов с последующим сбором информации в файл».

Таблица – Выходные данные

| № | Поле/объект | Тип данных | Ограничения |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 1 | Logs.text | Файл | Обязательно к заполнению |
| 2 | Снашоты | ComboBox | Обязательно к заполнению |
| 3 | Фотография графика результатов | Png | Не обязательно к заполнению |

## Методы

Данная программа написана с использованием различных парадигм объектно-ориентированного программирования, таких как инкапсуляция, наследование и полиморфизм, чтобы обеспечить эффективность и гибкость разработки.

Использование инкапсуляции позволит ограничить доступ одних компонентов программы к другим, обеспечивая тем самым безопасность и конфиденциальность данных. Компоненты программы будут взаимодействовать только через определенные интерфейсы, что позволит контролировать доступ к внутренним ресурсам и методам.

Наследование является важным аспектом объектно-ориентированного программирования, позволяющим повторно использовать код и структуру программы. Благодаря наследованию, классы могут наследовать свойства и методы от других классов, что способствует упрощению разработки и обслуживания кода, а также позволяет независимо расширять программное обеспечение и интерфейсы.

Использование статического полиморфизма позволяет обрабатывать различные типы данных с помощью одного и того же интерфейса или метода. Это обеспечивает удобство и гибкость взаимодействия с данными различных типов, что позволяет эффективно управлять разнообразными ситуациями в программе.

Комбинирование этих парадигм объектно-ориентированного программирования позволит создать высокоэффективную, гибкую и легко расширяемую программу, способную эффективно обрабатывать данные и решать разнообразные задачи.

## Тесты

Для данного проекта рекомендуется использовать ручное тестирование. Данный метод позволит достичь более высокой точности и детализации при исследовании функциональности и интерфейса разрабатываемого приложения, что необходимо для приложения, которое работает с виртуальной средой и автоматизирует тестирование другого программного обеспечения.

Было выбрано функциональное тестирование белого ящика. Подробные сценарии и результаты тестовых испытаний для приложения сбора файлов журнала, запуска автоматизированных тестов представлены в «Приложении Б. Сценарий и результаты тестовых испытаний».

При проведении функционального тестирования белого ящика была применена методология, которая позволяет исследовать внутреннюю структуру приложения, а также проверить корректность работы его компонентов и подсистем. Данный подход позволяет осуществить детальную проверку функциональных возможностей приложения на уровне исходного кода.

## Проектирование

## Инфологическая модель

На рисунке 2 представлена инфологическая модель базы данных.

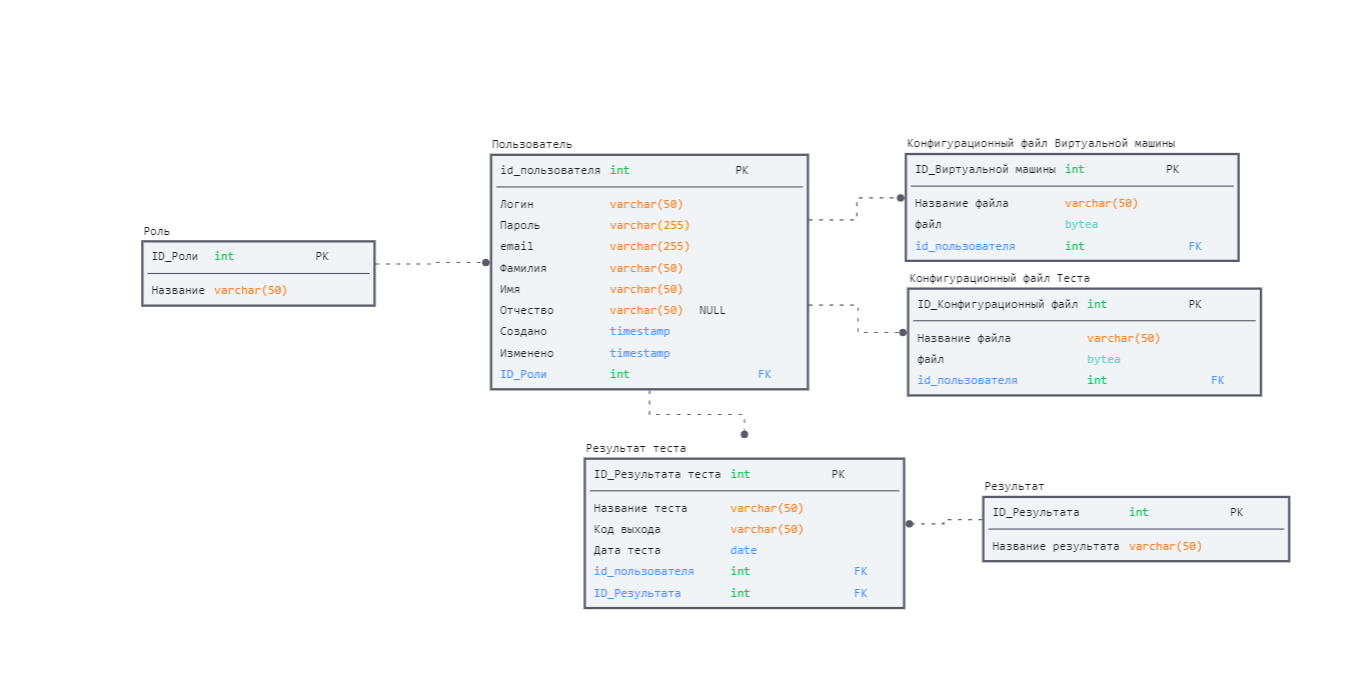


Рисунок – Инфологическая модель базы данных

## Даталогическая модель

На рисунке 3 представлена даталогическая модель базы данных.

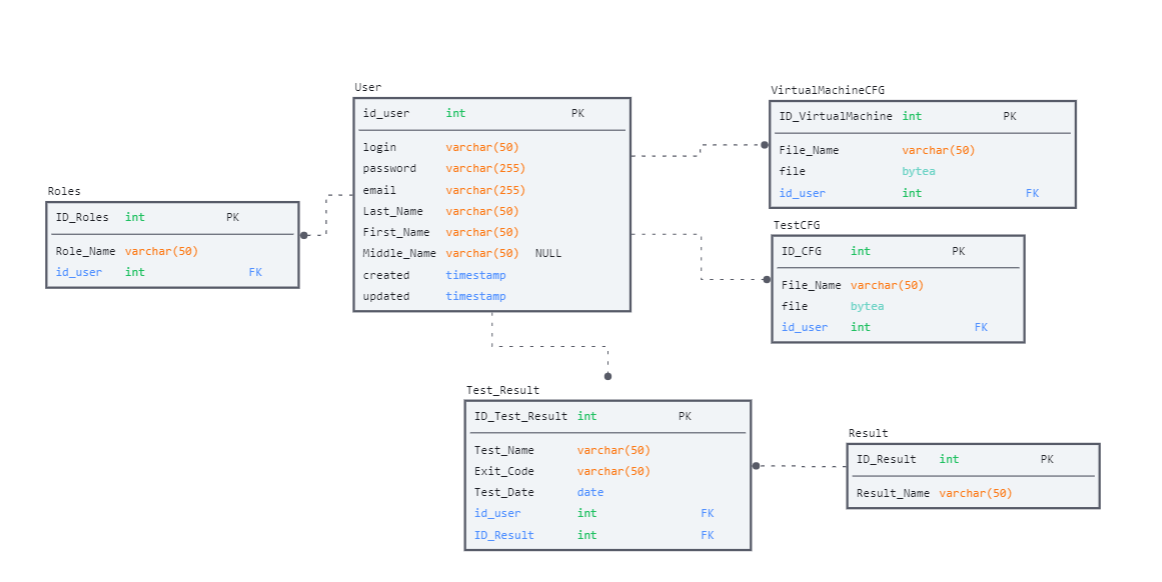


Рисунок – Даталогическая модель базы данных

## Словарь данных

В таблице 6 представлен словарь данных информационной системы «по управлению и запуску виртуальных машин и автоматизированных тестов с последующим сбором информации в файл».

Таблица – Словарь данных

| Ключ | Наименование | Тип данных | Обязательность  заполнения | Назначение |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| Role – таблица с ролями | | | | | |
| PK | ID\_Role | Int | Not null | Идентификатор роли  Auto increment |
|  | Role\_Name | Varchar(30) | Not null | Наименование  роли |
| FK | ID\_Profile | Int | Not null | Идентификатор пользователя |
| Profile – таблица с профилями | | | | | |
| PK | ID\_User | Int | Not null | Идентификатор профиля  Auto increment |
|  | Email | Varchar(255) | Not null | Логин пользователя |
|  | Password | Varchar(255) | Not null | Пароль пользователя |
|  | Last\_Name | Varchar(50) | Not null | Фамилия пользователя |
|  | Firtst\_Name | Varchar(50) | Not null | Имя пользователя |
|  | Middle\_Name | Varchar(50) | Null, default(‘-’) | Отчество пользователя |
|  | Created | TimeStamp | Not null | Время, когда была создана запись |
|  | Updated | TimeStamp | Not null | Время, когда была обновлена запись |
| VirtualMachineCFG | | | | | |
| PK | ID\_VirtualMachine | Int | Not null | Идентификатор конфигурационного файла виртуальной машины  Auto increment |
|  | File\_Name | Varchar(50) | Not null | Название файла |
|  | File | Bytea | Not null | Файл |
| FK | User\_ID | Int | Not null | Идентификатор профиля |
| TestCFG | | | | | |
| PK | ID\_CFG | Int | Not null | Идентификатор  Конфигурационного файла тестов  Auto increment |
|  | File\_Name | Varchar(50) | Not null | Название файла |
|  | File | Bytea | Not null | Файл |
| FK | User\_ID | Int | Not null | Идентификатор профиля |
| Result | | | | | |
| PK | ID\_Result | Int | Not null | Идентификатор результата  Auto increment |
|  | Result\_Name | Varchar(50) | Not null | Названия результата |
| Test\_Result | | | | | |
| PK | ID\_TestResult | Int | Not null | Идентификатор результата теста  Auto increment |
|  | Test\_Name | Varchar(50) | Not null | Название теста |
|  | Exit\_Code | Varchar(50) | Not null | Код выхода из программы |
|  | Test\_Date | Date | Not null | Дата тестирования |
| FK | User\_ID | Int | Not null | Идентификатор профиля |
| FK | ID\_Result | Int | Not null | Идентификатор результата |

## Структурная схема

На рисунке 4 представлена структурная схема информационной системы «по управлению и запуску виртуальных машин и автоматизированных тестов с последующим сбором информации в файл».

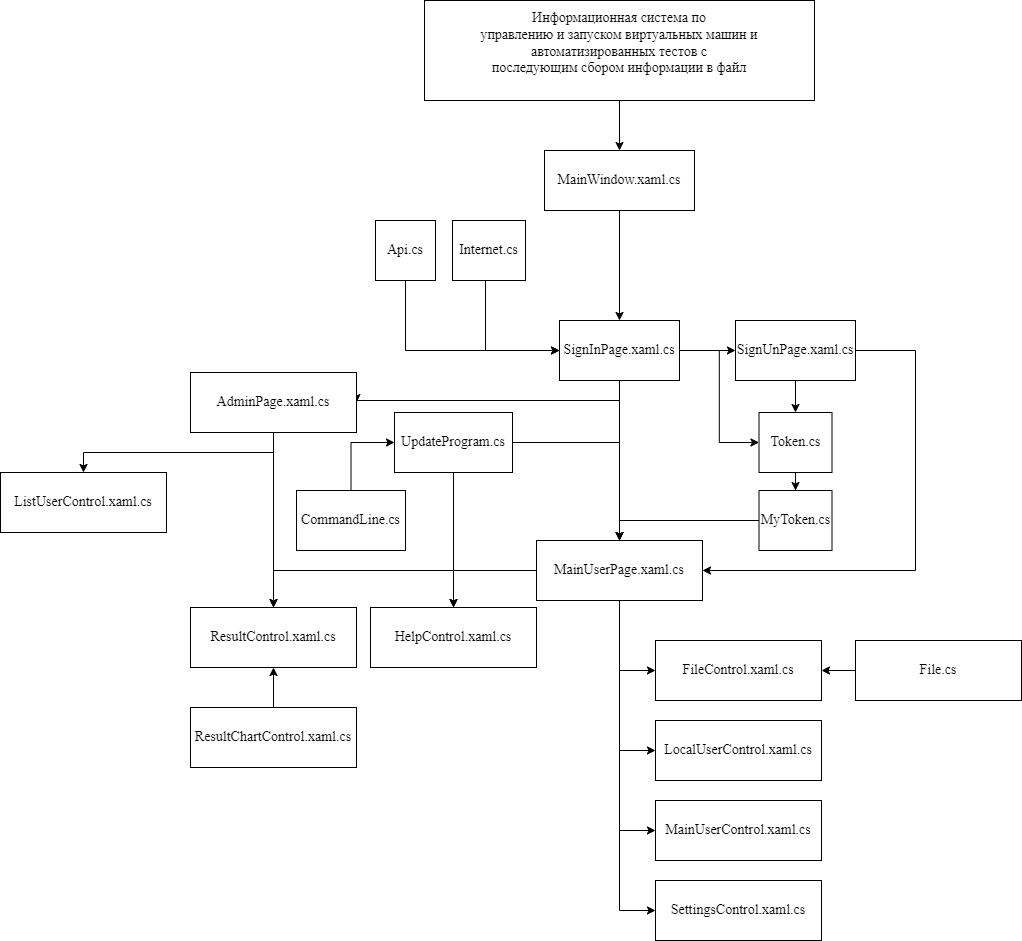


Рисунок – Структурная схема проекта

Таблица – Описание структурной схемы

|  |  |  |
| --- | --- | --- |
| № п/п | Наименование | Описание |
| 1 | 2 | 3 |
| 1 | MainWindows.xaml.cs | Главное окно, в котором отображаются все страницы |
| 2 | AdminPage.xaml.cs | Экран администратора, на которой отображаются необходимые вкладки |
| 3 | MainUserPage.xaml.cs | Главный экран тестировщика, на которой отображаются все вкладки |
| 4 | SignInPage.xaml.cs | Экран авторизации |
| 5 | SignUpPage.xaml.cs | Экран регистрации |
| 6 | MainUserControl.xaml.cs | Главная вкладка приложения, в которой можно включать, ставить на паузу, возобновлять, выключать, обновлять репозиторий, включать синхронизацию времени, выключать синхронизацию времени, выбирать снимок у виртуальных машин. Также добавлять и удалять запись тестов и запускать их. |
| 7 | LocalUserControl.xaml.cs | Вкладка локального запуска тестов, в которой можно добавлять и удалять запись тестов и запускать их. |
| 8 | SettingsControl.xaml.cs | Вкладка, на которой можно |
| 9 | ListUserControl.xaml.cs | Вкладка с таблицей пользователей, в котором присутствует функционал изменения, удаления пользователя |
| 10 | HelpControl.xaml.cs | Вкладка «Справка», в которой отображается версия программы, путь до исполняемого файла, функция обновления |
| 11 | FileControl.xaml.cs | Вкладка с функциями загрузки, удаления и скачивания файлов |
| 12 | ResultControl.xaml.cs | Вкладка с результатами тестов |
| 13 | ResultChartControl.xaml.cs | Вкладка с графиком результатов тестов |
| 14 | Api.cs | Класс, в котором хранится строка подключения к программному интерфейсу |
| 15 | CommandLine.cs | Класс с методом запуска процесса в командной строке |
| 16 | Internet.cs | Класс с методом проверки подключения к интернету |
| 17 | MyToken.cs | Статический класс, в котором хранится токен пользователя |
| 18 | UpdateProgram.cs | Класс с методом обновления программы |
| 19 | File.cs | Модель данных файла |
| 20 | Result.cs | Модель данных результата |
| 21 | TestResult.cs | Модель данных результат теста |
| 22 | Token.cs | Модель данных токена |
| 23 | User.cs | Модель данных пользователя |

## Функциональная схема

На рисунке 5 представлена функциональная схема приложения для сбора файлов журнала и запуска автоматизированных тестов.



Рисунок – Функциональная схема

## Диаграмма классов

На Рисунке 6 приставлена диаграмма классов приложения для сбора файлов журнала и запуска автоматизированных тестов. На ней графически изображены классы, которые состоят из полей, методов и свойств.

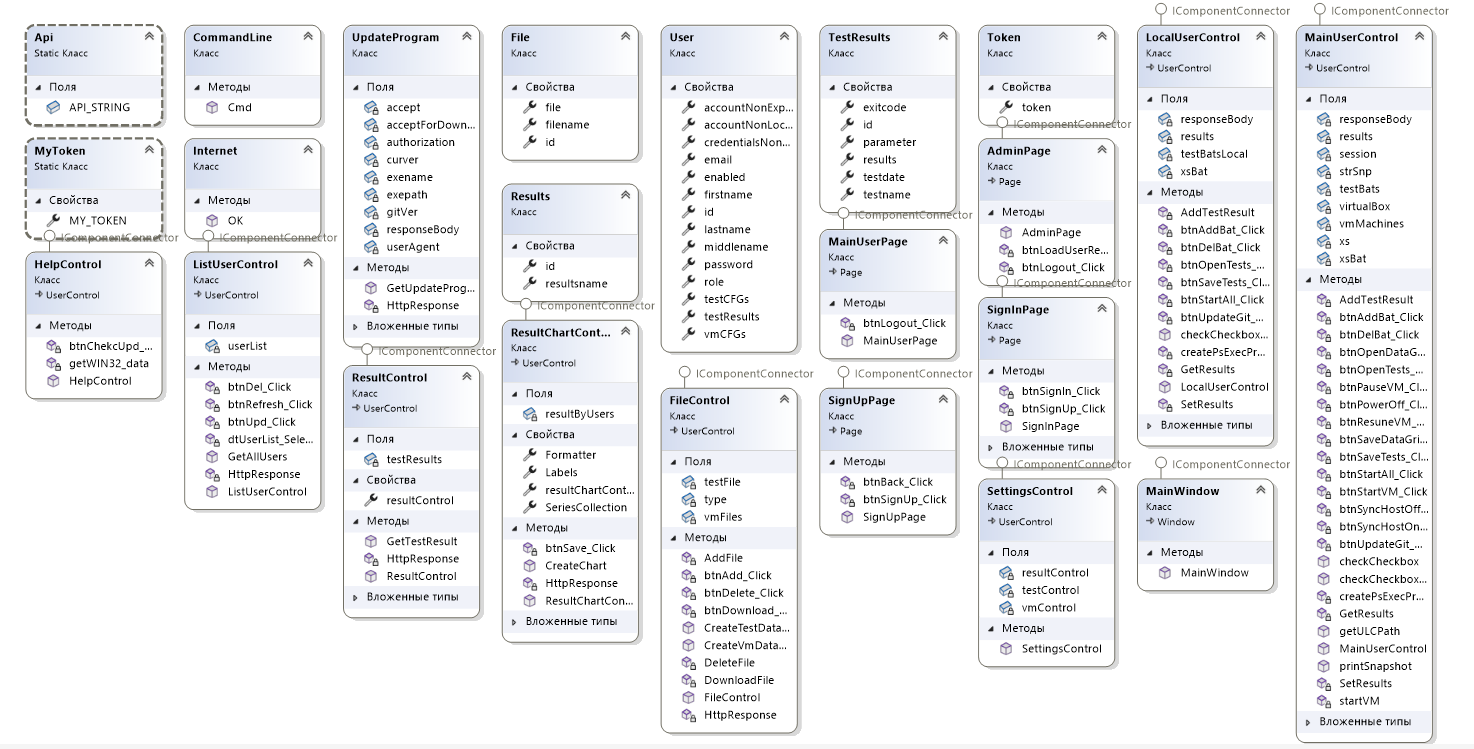


Рисунок – Диаграмма классов

## Схема тестирования

На рисунке 7 представлена схема тестирования, по которой проводилось тестирование приложения для сбора файлов журнала и запуска автоматизированных тестов.

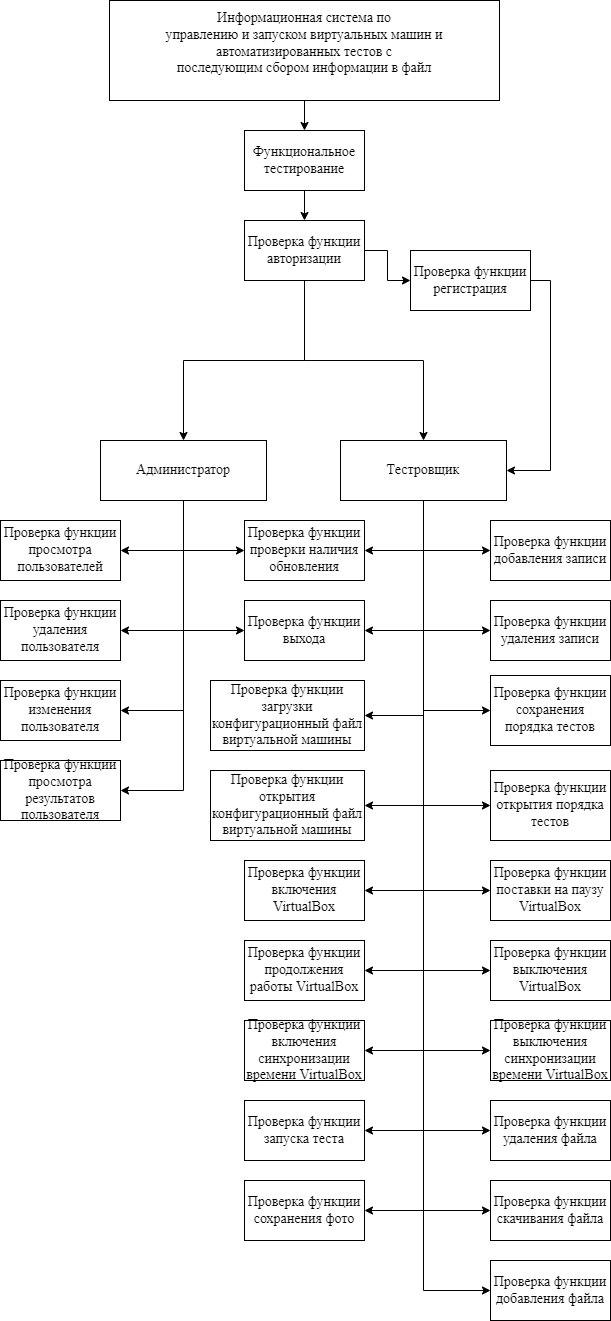


Рисунок – Схема тестирования

## Схема пользовательского интерфейса

На рисунке 8 представлена схема пользовательского интерфейса приложения для сбора файлов журнала и запуска автоматизированных тестов.



Рисунок – Схема пользовательского интерфейса

## Результат работы программы

Результатом работы программы является готовое программное обеспечение. Результат работы приведен на рисунке 9 Полное описание работы приложения находится в документе «Приложение В. Руководство пользователя».

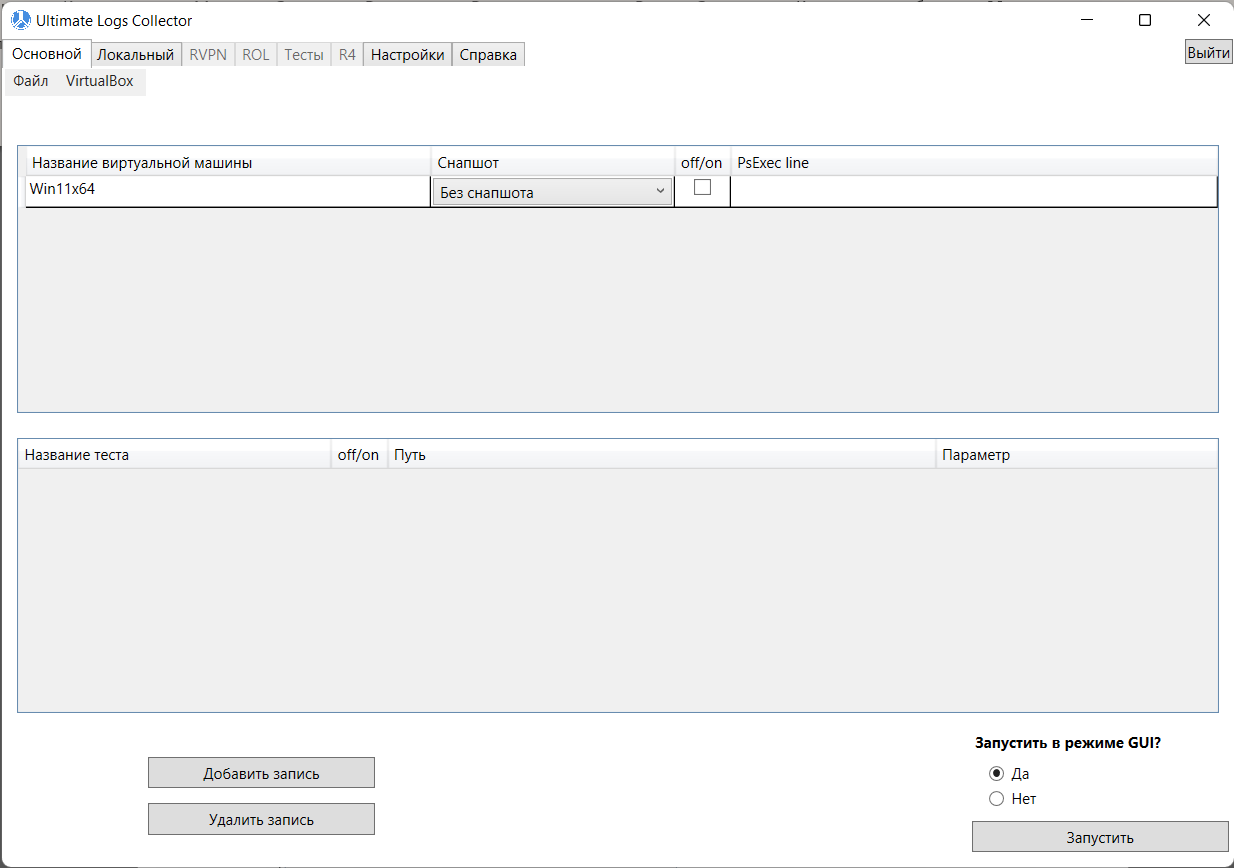


Рисунок – Результат работы приложения

# ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ

## Инструментальные средства

Для разработки информационной системы был выбран мощный и гибкий язык программирования C#, в связке с интегрированной средой разработки Visual Studio 2022. Эта среда разработки обладает широким набором инструментов и функциональных возможностей, что повышает продуктивность разработчиков. Благодаря быстрой прототипизации и простым, но мощным, инструментам отладки, разработка программы становится эффективной и удобной.

Для хранения данных в информационной системе была выбрана реляционная база данных PostgreSQL. Она отличается легким синтаксисом языка, что упрощает разработку и поддержку системы. Более того, PostgreSQL обладает расширяемостью и поддерживает выполнение сложных запросов, что позволяет эффективно работать с данными и обеспечивать высокую производительность системы. В качестве системы управления базами данных (СУБД) была выбрана PostgreSQL, которая является надежной и мощной платформой для работы с данными.

Для обеспечения быстрого и эффективного взаимодействия программы с базой данных, был разработан программный интерфейс с использованием фреймворка Spring. Он обладает простой конфигурацией и быстрым развертыванием, что упрощает процесс разработки и интеграции системы с базой данных. Благодаря этому программный интерфейс позволяет осуществлять быстрое и надежное взаимодействие с данными, обеспечивая высокую производительность и отзывчивость системы.

Для обеспечения изолированного окружения и простоты развертывания информационной системы, рекомендуется использование системы контейнеризации Docker, которая позволяет упаковывать приложение и его зависимости в контейнеры, что обеспечивает легкое масштабирование, управление и развертывание системы. С помощью Docker можно создать виртуальное окружение для информационной системы, где все необходимые компоненты, включая базу данных и программный интерфейс, будут работать в изолированной среде, обеспечивая стабильность и безопасность системы.

## Отладка программы

Отладка производилась с помощью встроенных инструментов Microsoft Visual Studio 2022, а именно точки остановы (Рисунок 10).

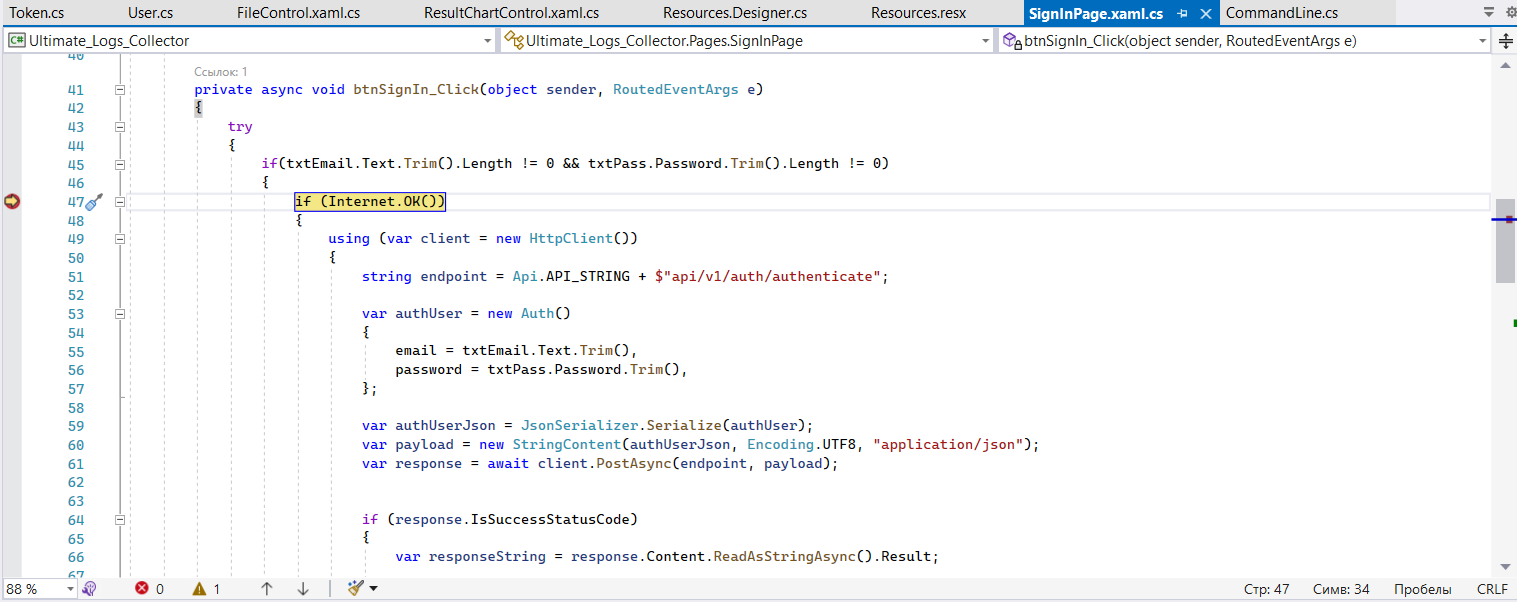


Рисунок – Точка остановы

Также использовалась проверка данных с подсказкой по данным (Рисунок 11).

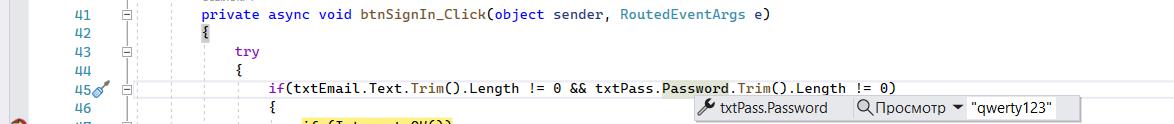


Рисунок – Проверка данных

Использовался инструмент для просмотра данных в переменных с использованием окон «Локальные» и «Видимые» (Рисунок 12).

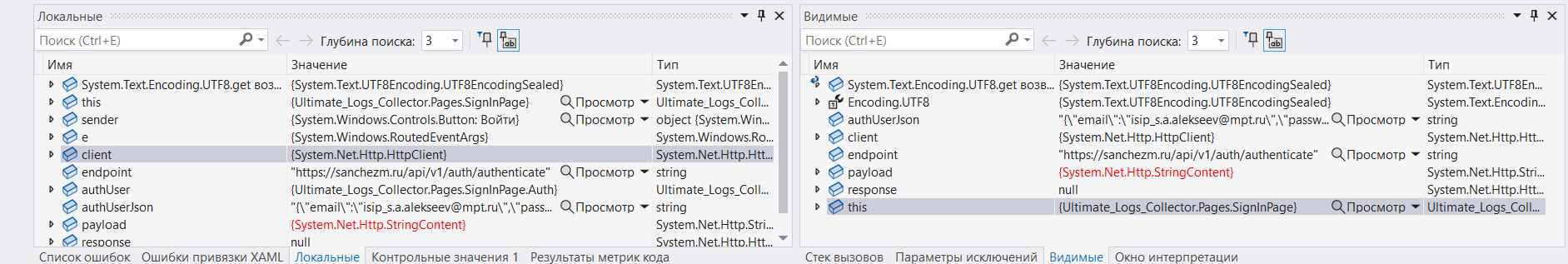


Рисунок – Окна «Локальные» и «Видимые»

Отладка программного интерфейса проводилась с помощью встроенного инструмента IntelliJ IDEA 2022.2.1 – точка остановы (Рисунок 13).

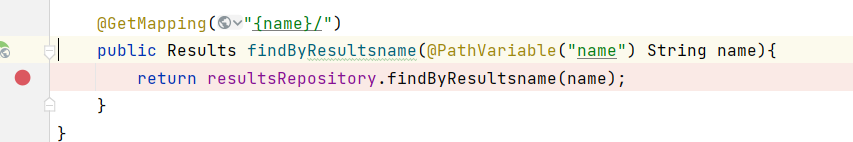


Рисунок – Точка остановы

## Защитное программирование

Было проведено защитное программирование с помощью блока обработчика исключений try-catch показанный на рисунке 14.



Рисунок – Обработчик исключений

Так же были добавлены механизмы обработки ошибок при некорректном вводе данных и вывод их пользователю продемонстрирован на рисунке 15.

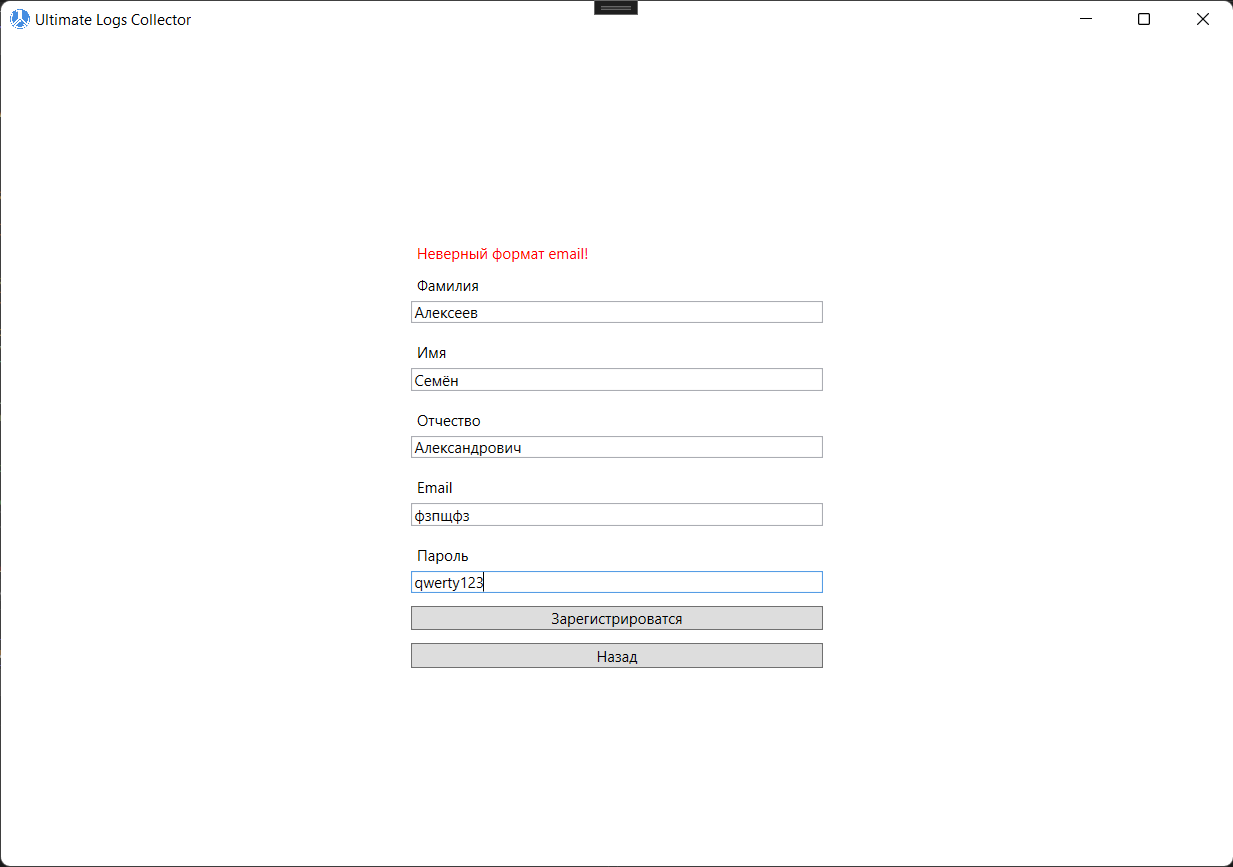


Рисунок – Защита от некорректных данных

## Характеристика программы

В таблице 8 представлена характеристика программы.

Таблица – Характеристика программы

| Номер | Модуль | Описание | Количество строк кода | Размер (в Кбайтах) |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| 1 | MainWindows.xaml.cs | Главное окно, в котором отображаются все страницы | 387 | 3 |
| 2 | MainWindow.xaml | Разметка главного окна | 235 | 21 |
| 3 | AdminPage.xaml.cs | Экран администратора, на которой отображаются необходимые вкладки | 289 | 15 |
| 4 | AdminPage.xaml | Разметка экрана администратора | 234 | 17 |
| 5 | MainUserPage.xaml.cs | Главный экран тестировщика, на которой отображаются все вкладки | 390 | 28 |
| 6 | MainUserPage.xaml | Разметка главного экрана тестировщика | 302 | 3 |
| 7 | SignInPage.xaml.cs | Экран авторизации | 205 | 16 |
| 8 | SignInPage.xaml | Разметка экрана авторизации | 167 | 2 |
| 9 | SignUpPage.xaml.cs | Экран регистрации | 202 | 26 |
| 10 | SignUpPage.xaml | Разметка экрана регистрации | 377 | 31 |
| 11 | MainUserControl.xaml.cs | Главная вкладка приложения, в которой можно включать, ставить на паузу, возобновлять, выключать, обновлять репозиторий, включать синхронизацию времени, выключать синхронизацию времени, выбирать снимок у виртуальных машин. Также добавлять и удалять запись тестов и запускать их. | 63 | 29 |
| 12 | MainUserControl.xaml | Разметка главной вкладки приложения | 284 | 12 |
| 13 | LocalUserControl.xaml.cs | Вкладка локального запуска тестов, в которой можно добавлять и удалять запись тестов и запускать их. | 211 | 6 |
| 14 | LocalUserControl.xaml | Разметка локальной вкладки | 90 | 26 |
| 15 | SettingsControl.xaml.cs | Вкладка, на которой можно просмотреть результаты тестов | 50 | 14 |
| 16 | SettingsControl.xaml | Разметка вкладки настроек | 289 | 17 |
| 17 | ListUserControl.xaml.cs | Вкладка с таблицей пользователей, в котором присутствует функционал изменения, удаления пользователя | 399 | 9 |
| 18 | ListUserControl.xaml | Разметка вкладки таблицы пользователей | 189 | 33 |
| 19 | HelpControl.xaml.cs | Вкладка «Справка», в которой отображается версия программы, путь до исполняемого файла, функция обновления | 33 | 17 |
| 20 | HelpControl.xaml | Разметка вкладки «Справка» | 324 | 6 |
| 21 | FileControl.xaml.cs | Вкладка с функциями загрузки, удаления и скачивания файлов | 172 | 11 |
| 22 | FileControl.xaml | Разметка вкладки работы с файлами | 78 | 12 |
| 23 | ResultControl.xaml.cs | Вкладка с результатами тестов | 211 | 14 |
| 24 | ResultControl.xaml | Разметка вкладки с результатами тестов | 91 | 16 |
| 25 | ResultChartControl.xaml.cs | Вкладка с графиком результатов тестов | 155 | 16 |
| 26 | ResultChartControl.xaml | Разметка вкладки с графиком результатов тестов | 56 | 35 |
| 27 | Api.cs | Класс, в котором хранится строка подключения к программному интерфейсу | 171 | 11 |
| 28 | CommandLine.cs | Класс с методом запуска процесса в командной строке | 100 | 19 |
| 29 | Internet.cs | Класс с методом проверки подключения к интернету | 349 | 11 |
| 30 | MyToken.cs | Статический класс, в котором хранится токен пользователя | 238 | 3 |
| 31 | UpdateProgram.cs | Класс с методом обновления программы | 124 | 25 |
| 32 | File.cs | Модель данных файла | 91 | 18 |
| 33 | Result.cs | Модель данных результата | 136 | 31 |
| 34 | TestResult.cs | Модель данных результат теста | 41 | 19 |
| 35 | Token.cs | Модель данных токена | 314 | 10 |
| 36 | User.cs | Модель данных пользователя | 122 | 4 |

# ЗАКЛЮЧЕНИЕ

В заключении хочется отметить, что реализация выбранного дипломного проекта на тему «Разработка системы управления тестированием и мониторингом виртуальной среды (на примере ООО «Фаматек»)» значительно расширила мой практический опыт в работе с ведущими инструментами разработки, такими как Visual Studio 2022, pgAdmin 4, IntelliJ IDEA 2022, а также позволила приобрести опыт работы с API, GitHub и Docker.

В ходе реализации данного дипломного проекта успешно была разработана информационная система, основанная на языке программирования C# с использованием системы построения клиентского приложения WPF. Эта система предоставляет широкие возможности для управления и запуска виртуальных машин, а также автоматизированных тестов с последующим сбором информации в файл. Главная цель системы заключается в упрощении процесса тестирования программных продуктов на виртуальных машинах.

Используя разработанное приложение, мы заметили, что основной сценарий программы работает без сбоев и значительно упрощает работу отдела тестирования. В рамках приложения пользователи имеют возможность регистрироваться, авторизироваться, а также добавлять, изменять, удалять, просматривать данные и графики, а также скачивать файлы. Однако, стоит отметить некоторые существенные недостатки приложения. В частности, приложение может зависать при авторизации, регистрации и запуске тестов на виртуальных машинах. Кроме того, вкладка с результатами тестов не обновляется автоматически, что требует ручного обновления. Дополнительно, приложение не поддерживает более новые версии VirtualBox. Также, при добавлении файла, если нажать кнопку "Отмена", будут добавлены пустые данные, что является недопустимым.

В целом, выполнение дипломного проекта позволило расширить мои знания и опыт в разработке информационных систем, а также выявить некоторые области, в которых требуется улучшение и доработка приложения. Продолжение работы над системой позволит устранить выявленные недостатки и создать более полноценное и надежное решение для управления и тестирования виртуальной среды.

# ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ

1. ГОСТ 19404- 79 ЕСПД. Пояснительная записка. ПЕРЕИЗДАНИЕ Января 2010 г.
2. ГОСТ 19.101-77 ЕСПД. Виды программ и программных документов.
3. ГОСТ 19.103-77 ЕСПД. Обозначение программ и программных документов.
4. ГОСТ 19.105-78 ЕСПД. Общие требования к программным документам.
5. ГОСТ 7.80-2000 СИБИД. Библиографическая запись. Заголовок. Общие требования и правила составления.
6. Адитья Бхаргава. Грокаем алгоритмы. Издательский дом «Питер».
7. Электронная свободная энциклопедия [Электронный ресурс] URL: https://ru.wikipedia.org/wiki/Microsoft\_Visual\_Studio (дата обращения 15.11.2022).
8. Электронная свободная энциклопедия [Электронный ресурс] URL: https://ru.wikipedia.org/wiki/Microsoft\_Word (дата обращения 15.11.2022).
9. Электронная свободная энциклопедия [Электронный ресурс] URL: https://ru.wikipedia.org/wiki/Windows\_11 (дата обращения 15.11.2022).
10. Прайс Марк. C# 8 и .NET Core. Разработка и оптимизация. Издательский дом «Питер».
11. Документация по Windows Presentation Foundation [Электронный ресурс] URL: https://docs.microsoft.com/ru-ru/dotnet/desktop/wpf/?view=netdesktop-6.0 (дата обращения 02.02.2022).
12. Сайт, посвященный структурированию всей информации о бизнес- приложениях [Электронный ресурс] URL: https://app.diagrams.net (дата обращения 19.01.2022).
13. Роберт Мартин, Мика Мартин. Принципы, паттерны и методики гибкой разработки на языке C#.
14. About repositories | GitHub https://docs.github.com/en/repositories/creating-and-managing-repositories/about-repositories (дата обращения 05.12.2022)
15. About pull requests | GitHub https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/proposing-changes-to-your-work-with-pull-requests/about-pull-requests (дата обращения 05.12.2022)
16. Creating a personal access token | GitHub https://docs.github.com/en/authentication/keeping-your-account-and-data-secure/creating-a-personal-access-token (дата обращения 05.12.2022)
17. Дж Дрейк и Дж Уорсли. PostgreSQL. Для профессионалов. Издательский дом «Питер».
18. Линн Бейли. Изучаем SQL. Издательский дом «Питер».
19. Build Restful API With Spring Boot using PostgreSQL and Spring Data JPA | Full Course for Beginner | https://www.youtube.com/watch?v=ZTxn38j4DJE (дата обращения 10.10.2022)
20. Http Get And Post In C# | YouTube https://www.youtube.com/watch?v=Yi-O-HBGPeU (дата обращения 11.11.2022)