МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»
Кафедра дискретной математики и информационных технологий

РАЗРАБОТКА ПРИЛОЖЕНИЯ С MVVM АРХИТЕКТУРОЙ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 421 группы
направления 09.03.01 — Информатика и вычислительная техника
факультета КНиИТ
Серова Андрея Николаевича

Научный руководитель
старший преподаватель _____ М. В. Белоконь

Заведующий кафедрой
к. ф.-м. н., доцент _____ Л. Б. Тяпаев

Саратов 2024

ВВЕДЕНИЕ

Десктопное приложение – программа, которая устанавливается на компьютер пользователя и работает под управлением операционной системы. В современном мире десктопные приложения остаются неотъемлемой частью повседневной жизни как в профессиональной сфере, так и для личного пользования. Несмотря на стремительное развитие веб-технологий и мобильных приложений, десктопные решения все еще широко используются, особенно в тех областях, где требуется высокая надежность работы, возможность работы без выхода в интернет и интеграция с локальными ресурсами. Многие предприятия и профессиональные пользователи по-прежнему отдают предпочтение десктопным приложениям для выполнения сложных вычислений, работы с большими объемами данных и выполнения специализированных задач.

За основу данной работы был взят проект готовой экономической модели формирования земельной ренты (МФЗР), созданный для НИИ экономики и организации агропромышленного комплекса. Команде, разрабатывающей данный проект, необходим инструмент, который позволит эффективно анализировать и обрабатывать данные, собранные в рамках модели МФЗР. Это включает в себя не только возможность визуализации и редактирования таблиц, но и применение различных методов и алгоритмов. Для этого требуется приложение со сложной архитектурой, имеющее потенциал к масштабированию, и понятным интерфейсом, а также позволяющее хранить и использовать необходимые для пользователя скрипты.

Целью данной работы является разработка десктопного приложения — редактора таблиц, который обеспечивал бы удобное управление и редактирование таблиц с возможностью использования скриптов для выполнения различных операций над данными.

Для достижения поставленной цели необходимо решить следующие задачи:

- проанализировать и изучить необходимые технологии;
- выбрать архитектуру приложения;
- узнать современные подходы к разработке приложений;
- разработать приложение.

Бакалаврская работа состоит из введения, 3 разделов, а именно: «Ана-

лиз предметной области», «Программная реализация», «Обзор приложения», заключения и списка использованных источников. Общий объём работы – 52 страницы, из них 52 страниц – основное содержание, включая 22 рисунка, список использованных источников – 24 наименования.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

В первом разделе «Анализ предметной области» рассказано о текущих инструментах и технологиях, используемых командой, создающей проект готовой экономической модели формирования земельной ренты. Также перечислены основные трудности, возникающие при работе с существующим инструментарием. Разрабатываемое в данной работе приложение предполагалось обеспечить функционалом, улучшающим пользовательский опыт в контексте перечисленных трудностей в начале раздела.

Большую часть раздела составляет описание выбора подходящих под задачу технологий. В качестве языка программирования был выбран C#, благодаря своим широким возможностям при разработке десктопных приложений.

При выборе фреймворка, на основе которого разрабатывалось приложение, постепенно исключались популярные решения, пока не осталось одно наиболее подходящее под задачи проекта. Так, после исключения других решений, оптимальным вариантом для разработки приложения остался WPF. Этот фреймворк обеспечивает возможность построения сложной архитектуры приложения, при этом поддержка XAML позволила сделать интерфейс удобным и привлекательным для пользователя.

В качестве библиотеки для работы с таблицами Excel была выбрана библиотека EPPlus, так как данная библиотека имеет хорошую производительность при работе с большими данными, что было необходимо для загрузки и обработки нескольких десятков таблиц одновременно.

Для изучения особенностей архитектуры приложений были выбраны и описаны популярные для десктопных приложений архитектуры. Архитектура MVVM является рекомендованным и широко применяемым шаблоном для разработки приложений на платформе WPF, поэтому для разработки приложения была выбрана именно эта архитектура.

После выбора архитектуры были описаны конкретные подходы к совместной работе фреймворка и архитектуры, такие как:

- Привязка данных.
- Интерфейс INotifyPropertyChanged.
- Команды.

В данном разделе также были описаны современные подходы к разра-

ботке приложений:

- Принципы SOLID: набор принципов, направленных на улучшение качества кода.
- Внедрение зависимостей: техника, позволяющая уменьшить связность компонентов и упростить их тестирование.
- Асинхронное программирование: подход, позволяющий выполнять задачи в фоне, не блокируя основной поток выполнения.
- Рефлексия в C#: механизм получения информации о структуре программы во время выполнения.

В конце раздела описывается тема "Дизайн и адаптивная верстка". В ней рассказывается о языке разметки XAML, который используется для создания пользовательского интерфейса в WPF, а также об адаптивной верстке в WPF, представляющей из себя методы и инструменты для создания интерфейсов, корректно отображающихся на устройствах с различными разрешениями экрана и размерами окон.

Во втором разделе «Программная реализация» рассказано о процессе разработки приложения, приведены конкретные этапы реализации.

Разработка началась с создания макетов основных разделов приложения, таких как главная страница, страница редактора, страница загрузчика, страница инструментов, страница справки и страница настроек. Макеты служили основой для последующей верстки интерфейса приложения.

Для лучшей организации кода, архитектурные части и дополнительные компоненты приложения были разделены на отдельные каталоги. Также позволило организовать архитектурный шаблон MVVM.

Также был приведен код использования Dependency Injection (DI) для управления зависимостями между объектами в приложении. Это позволило сделать систему более гибкой к тестированию и замене компонентов.

В том числе, в разделе было приведено описание того, как различные части приложения взаимодействуют друг с другом. Для обеспечения эффективного взаимодействия и обновления данных в приложении используется три основных подхода: привязка данных, интерфейс INotifyPropertyChanged и команды.

Описана реализация системы навигации через создание собственного элемента управления, позволяющего пользователям легко перемещаться меж-

ду различными разделами приложения. Приведен код для определения и настройки навигационных маршрутов и механизм обработки переходов.

Далее в работе приведён класс, созданный для обработки и конвертации данных из файлов Excel в формат, используемый приложением. В нем присутствуют методы загрузки и сохранения пользовательских данных.

Показана легко расширяемая система исполнения скриптов, построенная через взаимодействие классов с использованием делегатов. В качестве описания возможности расширяемости системы приведена последовательность действий по добавлению новых скриптов, а так же описан алгоритм работы системы с точки зрения программного кода.

В третьем разделе «Обзор приложения» рассказано о том, как пользователь взаимодействует с приложением. Первоначально пользователь видит начальную страницу, на которой предлагается обратиться к разделу "О приложении"для ознакомления с руководством. Начальная страница включает кнопки на боковой панели, позволяющие перемещаться между различными разделами приложения.

Переходя в раздел "О приложении пользователь получает описание приложения, инструкции по его использованию и контактную информацию разработчика. Этот раздел сопровождается соответствующими изображениями интерфейса.

Автоматическая загрузка таблиц происходит при запуске приложения, при условии наличия указанного пути к папке с таблицами. В случае необходимости изменения этого пути, пользователь может сделать это на странице настроек.

Во время загрузки таблиц кнопка "синхронизации с папкой"на странице загрузчика выделена серым цветом, что указывает на активную загрузку таблиц. Процесс загрузки реализован асинхронно, чтобы интерфейс оставался отзывчивым.

После завершения загрузки таблицы отображаются в виде карточек, которые можно добавлять в быстрый доступ. Пользователь может выбирать таблицы из верхней части экрана или удалять их из быстрого доступа с помощью круглой кнопки "удалить".

На странице приложения также доступен поиск по таблицам. Пользователь вводит имя таблицы для поиска, а результаты отображаются в соот-

ветствующем поле.

Описывается функционал редактора, где при выборе таблицы из быстрого доступа пользователь может редактировать её содержимое и сохранять изменения.

Далее в разделе показана страница инструментов и примеры скриптов, расположенных на ней. Показаны и описаны следующие скрипты:

1. Сложить ячейки – скрипт складывает ячейки в заданном диапазоне и выводит результат пользователю. Это демонстрирует возможность отклика на данные пользователя и может включать комплексные структуры, такие как кнопки, таблицы, графики.
2. Очистить ячейки – этот скрипт удаляет содержимое ячеек в указанном диапазоне, демонстрируя возможность редактирования таблиц через скрипты. Скрипты могут изменять данные, объединять таблицы или менять их структуру.
3. Запустить файл – позволяет запускать внешние исполняемые файлы формата .exe. Это расширяет функциональность приложения, позволяя интегрировать его с другими системами и алгоритмами для обработки таблиц.

Для взаимодействия со скриптами пользователю нужно перейти на соответствующую страницу, выбрать необходимый скрипт, ввести параметры и вызвать скрипт.

ЗАКЛЮЧЕНИЕ

В ходе выполнения дипломной работы было разработано приложение – редактор таблиц, который обеспечивает удобное управление таблицами и их редактирование таблиц, имеющий механизм создания и использования скриптов для выполнения различных алгоритмов.

Приложение построено на архитектуре MVVM, что обеспечивает четкое разделение логики, представления и данных, предоставляет возможности к тестированию и масштабированию. Были использованы современные подходы к разработке, такие как внедрение зависимостей, асинхронное программирование, что позволило сделать приложение более эффективным и поддерживаемым.

В качестве основы приложения был использован фреймворк WPF, благодаря чему получилось организовать полноценную MVVM архитектуру через инструменты привязки и команды, а так же создать интуитивно понятный интерфейс с использованием языка разметки XAML.

Создана масштабируемая основа для хранения и применения пользовательских скриптов для операций над таблицами, что позволяет приложению быть удобным и функциональным инструментом для пользователя.

Основные источники информации:

1 Албахари, Н. C# 7.0. Справочник. Полное описание языка / Н. Албахари. – Москва, Санкт-Петербург, Россия: Диалектика, 2018.

2 Евдокимов, В. C# на примерах / В. Евдокимов. – Санкт-Петербург, Россия: Наука и техника, 2019.

3 Что такое MVVM архитектура. [Электронный ресурс]. — URL: https:// apptractor.ru/info/articles/chto-takoe-mvvm-arhitektura.html (Дата обращения 12.05.2024). Загл. с экр. Яз. рус.

4 Введение в язык XAML. [Электронный ресурс]. — URL: https:// metanit .com/sharp/wpf/2.php (Дата обращения 12.05.2024). Загл. с экр. Яз. рус.