МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
Некоммерческое акционерное общество
«АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ»
Кафедра IT-инжиниринг

**ДОПУЩЕН К ЗАЩИТЕ**
Заведующий кафедрой
PhD, доцент
_____ Т.С. Картбаев
«____» _____ 2019 г.

# ДИПЛОМНЫЙ ПРОЕКТ

На тему: Разработка веб-конвертера по переводу казахских текстов с кириллицы на латиницу

Специальность 5В070300 – «Информационные системы»

Выполнил Геник М.А. Группа ИСу-16-3

Научный руководитель _____ ст. преп. Купарова А.Т.
«13» 05 2019 г.

Консультанты:
по экономической части: к.э.н., доцент _____ А.И.Бекишева
«03» 05 2019 г.

по безопасности жизнедеятельности: д.т.н., ст.преп _____ Ш.Ш.Бекбасаров
«14» 05 2019 г.

Программное обеспечение: ст. преп. _____ М.Н.Майкотов
«14» 05 2019 г.

Нормоконтролер: PhD, ст. преп. _____ Ж.К.Алимсеитова
«14» 05 2019 г.

Рецензент: к.т.н., зав. каф. «Информационная безопасность» КазНИТУ
_____ Н.А.Сейлова
«____» _____ 2019 г.

Алматы 2019

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
Некоммерческое акционерное общество
«АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ»

Институт систем управления и информационных технологий
Кафедра IT-инжиниринг
Специальность 5В070300 – Информационные системы

**Задание**
на выполнение дипломного проекта

Студенту Геник Максиму Андреевичу
Тема проекта: Разработка веб-конвертера по переводу казахских текстов с кириллицы на латиницу
Утверждена приказом по университету № 124 от « 26 » октября 2019 г.
Срок сдачи законченного проекта « 20 » декабря 2019 г.

Исходные данные к проекту, требуемые параметры результатов проектирования и исходные данные объекта:
а) универсальный инструмент для создания прототипов «Justinmind»;
б) среда разработки Visual Studio для создания программной части;
в) офисный пакет приложений Microsoft Office;
г) консоль запросов для проведения скоростных тестов приложения.
Перечень вопросов, подлежащих разработке в дипломном проекте, или краткое содержание дипломного проекта:
а) область применения разработки;
б) описание исходной ситуации;
в) выбор средств и технологий для решения поставленных задач;
г) моделирование и проектирование веб-приложения, программных и аппаратных средств;
Перечень графического материала (с точным указанием обязательных чертежей): Представлены 14 таблиц, 51 иллюстрация.
Основная рекомендуемая литература:
1. James Chambers, David Paquette, Simon Timms ASP.NET Core Application Development: Building an application in four sprints (Developer Reference), 2016;
2. Andrew Lock ASP.NET Core in Action, 2018;
3. Michael Fitzgerald Introducing Regular Expressions, 2012;
4. Джозеф А., Бен А. C# 6.0. Карманный справочник, 2015;
5. Mark J. Price C# 6 and .NET Core 1.0: Modern Cross-Platform Development, 2016.

Консультация по работе (проекту) с указанием относящихся к ним разделов работы (проекта)

| Раздел | Консультант | Сроки | Подпись |
|---|---|---|---|
| Экономическая часть | А.И.Бекишева | 28.02 - 03.05.19 | |
| Безопасность жизнедеятельности | Ш.Ш.Бекбасаров | 26.02.19-14.05.19 | |
| Программное обеспечение | М.Н.Майкотов | 26.02.19 - 14.05.19 | |
| Нормоконтролер | Ж.К.Алимсеитова | 1.03.19 - 15.05.19 | |

График
подготовки дипломного проекта

| Наименование разделов, перечень разрабатываемых вопросов | Сроки представления научному руководителю | Примечание |
|---|---|---|
| Обзор существующего программного обеспечения | 01.01.2019-01.02.2019 | |
| Анализ технологий и возможных средств решения проблемы | 01.02.2019-14.02.2019 | |
| Графический интерфейс пользователя и Программная часть | 14.02.2019-30.04.2019 | |
| Экономическое обоснование разработки проекта | 30.04.2019-03.05.2019 | |
| Безопасность жизнедеятельности | 03.05.2019-14.05.2019 | |

Дата выдачи задания «14» января 2019 г.

Заведующий кафедрой _____ Т.С. Картбаев
(подпись)

Научный руководитель работы (проекта) _____ ст. преп. А.Т. Купарова
(подпись)

Задание принял к исполнению студент _____ М.А. Геник
(подпись)

## Аннотация

Тема: «Разработка веб-конвертера по переводу казахских текстов с кириллицы на латиницу».

Целью дипломного проекта является разработка веб-приложения для перевода казахских текстов с кириллицы на латиницу. При этом основная задача, выполняемая в данном дипломном проекте – разработать быстрый функционал по латинизации документов.

Дипломный проект состоит из 4 разделов. Введение раскрывает актуальность, цели и задачи разработки. Первый раздел содержит анализ существующего ПО, постановку задачи, а также обоснование выбора средств и технологий. Второй раздел содержит этапы разработки ПО. Третий раздел – экономическое обоснование разработки проекта. Четвертый раздел посвящен рассмотрению мероприятий по охране труда.

## Аңдатпа

Тақырыбы: «Қазақ мәтіндерін кириллицадан латын тіліне аудару бойынша веб-конвертор құру».

Бұл дипломдық жоба қазақ тіліндегі мәтіндерді кириллицадан латын тіліне аудару үшін веб-конвертерді жобалауға, әзірлеуге және енгізуге арналған. Бұл жағдайда, диплом жобасында орындалатын негізгі тапсырма құжаттарды төңкеру үшін жылдам функционалдылықты дамыту болып табылады.

Дипломдық жоба 4 бөлімнен тұрады. Кіріспе дамудың өзектілігі, мақсаттары мен міндеттерін көрсетеді.

Бірінші бөлімде қолданыстағы бағдарламалық қамтамасыз етуді талдау, мәселені тұжырымдау және құралдар мен технологияларды іріктеудің негіздемесі. Екінші бөлім бағдарламалық жасақтаманың даму кезеңдерін қамтиды. Үшінші бөлім жобаны әзірлеуге арналған бизнес-жағдай болып табылады. Төртінші секция еңбекті қорғау жөніндегі шараларды қарауға арналған.

## Annotation

Theme: “Development of a web-converter for translating Kazakh texts from Cyrillic to Latin”.

The aim of the thesis is to develop a web application for translating Kazakh texts from Cyrillic to Latin. In this case, the main task performed in this thesis project is to develop a fast functionality for document roping.

Thesis consists of 4 sections. The introduction reveals the relevance, goals and objectives of the development. The first section contains an analysis of the existing software, the formulation of the problem, and the rationale for the selection of tools and technologies. The second section contains the stages of software development. The third section is a business case for project development. The fourth section is devoted to the consideration of measures for labor protection.

## Введение

Актуальность и востребованность темы дипломного проекта тесно связаны с переходом казахского алфавита с кириллицы на латиницу.

Экс-президент государства Нурсултан Назарбаев утвердил своим указом новый алфавит казахского языка, который теперь основывается на латинской графике и подписал указ о мероприятиях по переводу алфавита с кириллицы на латиницу. Кроме того, министр образования и науки РК приказом от 27 апреля 2018 года утвердил график перехода казахского алфавита на латинский до 2025 года, охватывающий все уровни образования. А заместитель главы Министерства культуры Ерлан Кожагапанов сообщил, что в период с 2021 по 2023 года планируется латинизация документов на казахском языке [1].

Ожидаемым результатом и целью дипломного проекта является разработка веб-приложения для перевода казахских текстов с кириллицы на латиницу, которое находится в свободном доступе и может быть использовано любым пользователем глобальной сети интернет.

Практическая значимость дипломного проекта заключается в возможности применения её на практике с целью автоматизации процесса латинизации документов с казахским текстом.

В ходе выполнения дипломного проекта на первом этапе был проведен обзор существующего программного обеспечения и анализ их недостатков, после чего разработан пользовательский интерфейс веб-приложения, затем этап непосредственной реализации сервиса, и в заключительной части работы будет произведено тестирование и апробация программного продукта.

**Содержание**

# 1 Основная часть

## 1.1 Цель разработки и анализ ее использования

### 1.1.1 Обзор существующего программного обеспечения

На данный момент на просторах интернета можно найти не мало программ, сайтов и веб-приложений, посвященных переводу казахских текстов с кириллицы на латиницу. Их количество существенно увеличилось после выхода соответствующих законов и постановлений о латинизации казахского алфавита.

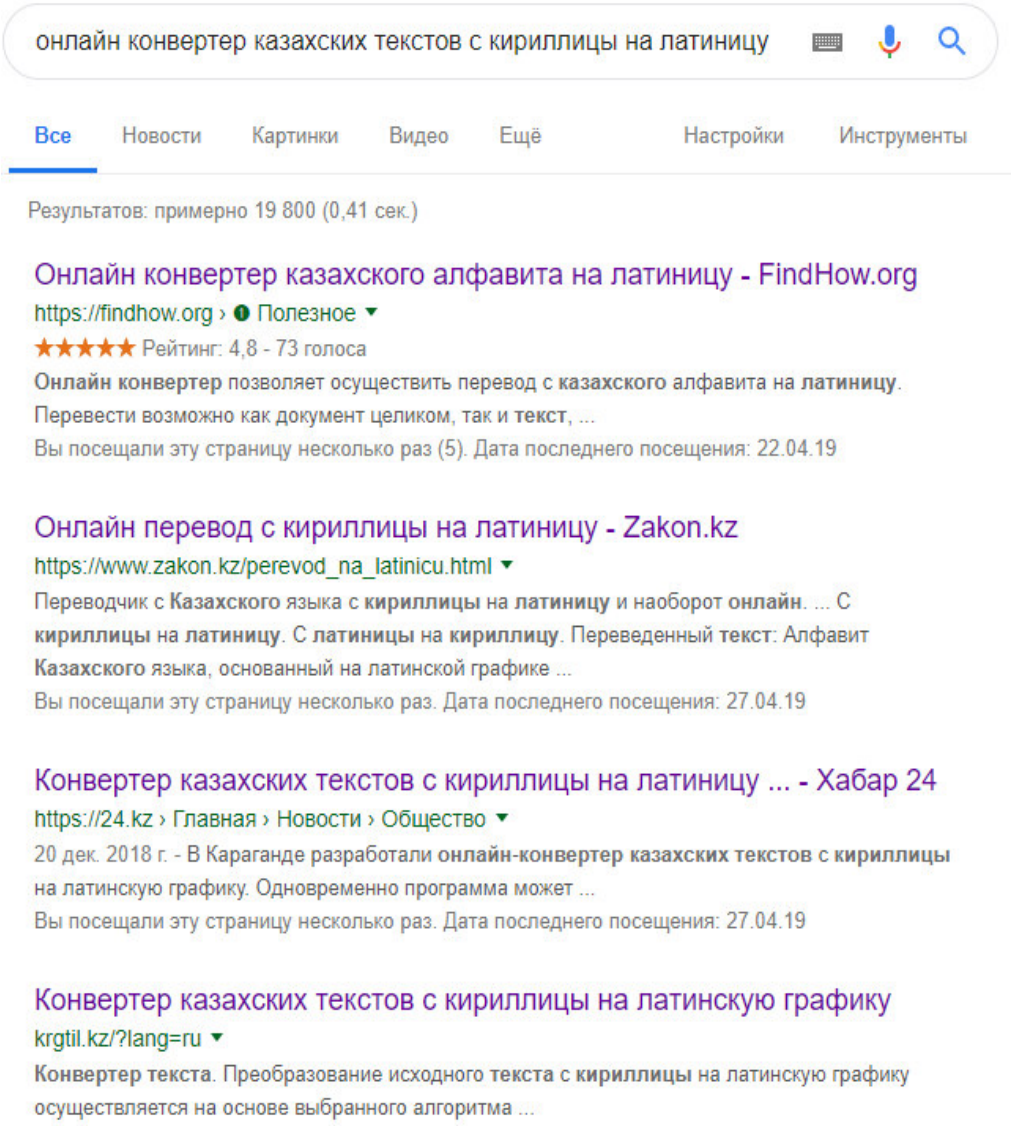


Рисунок 1.1 – Первая страница результата поискового запроса

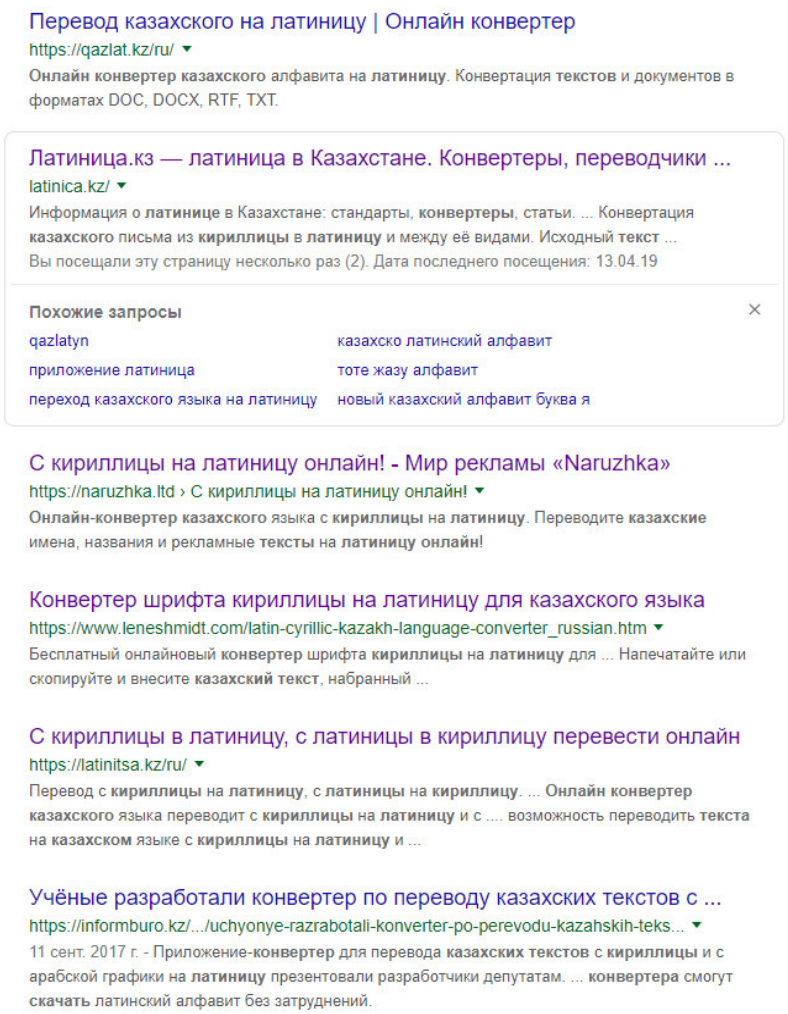


Рисунок 1.2 – Первая страница результата поискового запроса (продолжение)

На рисунках 1.1 и 1.2 видно, что поисковой запрос «онлайн конвертер казахских текстов с кириллицы на латиницу» выдает не мало результатов. На первой странице результата поискового запроса вы видим сайты: findhow.org, zakon.kz, 24.kz, krgtil.kz, qazlat.kz, latinica.kz, naruzhka.ltd, leneshmidt.com, latinitsa.kz, informburo.kz. Из них, услуги по латинизации предлагают 8 сайтов: findhow.org, zakon.kz, krgtil.kz, qazlat.kz, latinica.kz, naruzhka.ltd, leneshmidt.com, latinitsa.kz. Другие 2 сайта имеют лишь информативный характер и служат для размещения на них новостей по данной тематике.

Был проведен сравнительный анализ результатов запроса. Для того, чтобы результат сравнения был наиболее информативным и наглядным, были выделены некоторые признаки, по которым видно все недостатки и преимущества существующего программного обеспечения.

Основные признаки, по которым было проведено сравнение: функциональность, скорость работы, интерфейс, дополнительные особенности.

Из всех результатов, лишь один сайт предлагает услуги латинизации документов, в отличие от остальных, которые включают только функционал по конвертации отдельного текста.

Этим сайтом является findhow.org. На нем имеется функционал для латинизации казахского текста. Ограничение – 20000 знаков. А также возможность латинизации документов с расширением .docx, которые содержат казахский текст. Ограничение – максимальный размер файла 20МБ. Латинизация происходит сразу после загрузки документа, скачивание – после нажатия на кнопку «Скачать». Это значит, что, если пользователь случайно выбрал не тот документ, ему придется ждать, когда закончится предыдущий процесс латинизации и только потом произвести повторную попытку.

Для определения скорости выполнения латинизация документов использовались файлы Word с разным количеством страниц: 5, 10, 20, 50, 200, 500, 1500, 3000, 6000, 12500. А также инструменты разработчика для браузера Google Chrome: «Web developer» и «Performance-analyser».

В таблице 1.1 наглядно показаны результаты измерений.

Таблица 1.1 – Скорость выполнения латинизации документов

| Кол-во страниц | Время (чч:мм:сс:мл) |
|---|---|
| 5 стр. | 00:00:01:43 |
| 10 стр. | 00:00:01:46 |
| 20 стр. | 00:00:01:63 |
| 50 стр. | 00:00:01:95 |
| 200 стр. | 00:00:03:20 |
| 500 стр. | 00:00:05:76 |
| 1500 стр. | 00:00:15:95 |
| 3000 стр. | 00:00:30:08 |
| 6000 стр. | 00:01:02:04 |
| 12500 стр. | 00:02:04:37 |

Графический интерфейс сайта выполнен в стиле «минимализм». Страница представлена на одном языке – русский. Отсутствуют лишние элементы дизайна. Внимание пользователя сконцентрировано на функционале. Сайт является адаптивным и кросс браузерным, поэтому оптимально отображается на разных платформах и браузерах.

На основании анализа существующего программного обеспечения по конвертации казахских текстов с кириллицы на латиницу, можно выделить некоторые проблемы, решение которых необходимо и обязательно при разработке нового веб-приложения. Среди них:

– увеличение функциональности приложения, а именно добавление формата презентаций (.pptx);

– сокращение времени обработки файлов;

– разработка современного графического пользовательского интерфейса;

– расширение функционала;

– разработать мультиязычный ресурс.

### 1.2 Анализ технологий и возможных средств решения проблемы.

Широкий взгляд на выявленные проблемы позволяет обратить внимание на то, что есть не мало средств и технологий, позволяющих обрабатывать содержимое документов. Среди них языки программирования Javascript, C#, Python и другие.

В январе 2017 года компания Microsoft разработала библиотеку для работы с Office документами. Этот набор контента предоставляет документацию и руководство для строго типизированных классов в Open XML SDK 2.5 для Office. SDK построен на API-интерфейсе System.IO.Packaging и предоставляет строго типизированные классы для управления документами, которые соответствуют спецификации форматов файлов Office Open XML. Спецификация форматов файлов Office Open XML является открытым международным стандартом ECMA-376, второго издания и ISO / IEC 29500. Форматы файлов Open XML полезны для разработчиков, поскольку они являются открытым стандартом и основаны на известных технологиях: ZIP и XML.

Возможности данной библиотеки полностью отвечают требованиям для решения выявленных проблем, поэтому с наибольшей вероятностью будут использованы при реализации веб-конвертера казахских текстов с кириллицы на латиницу.

### 1.3 Выбор средств и технологий

Учитывая, что самый распространенный офисный пакет приложений – это Microsoft Office, веб-конвертер ориентирован именно на него. Этот пакет включает в себя программное обеспечение для работы с различными типами документов: текстами, электронными таблицами, презентациями и др.

Компания Microsoft предлагает очень обширный спектр программного и аппаратного обеспечения. Поэтому было бы абсолютно логично, для обработки файлов пакета приложений компании Microsoft использовать средства разработки, которые также поставляются этой компанией.

#### 1.3.1 Среда разработки Visual Studio

Visual Studio (сокр. VS), интегрированная среда разработки – это креативная стартовая панель, которую можно использовать для редактирования, отладки и сборки кода, а затем для публикации приложения [2]. VS – это программа, включающая в себя большое многообразие функций, которая может использоваться во многих направлениях разработки программного обеспечения. В дополнение к стандартному редактору и

отладчику, которые предоставляет большинство сред для разработки, Visual Studio имеет в своем арсенале компиляторы, инструменты завершения кода, графические дизайнеры и много других функций, которые облегчают задачу программиста разрабатывать программное обеспечение [3].

Эта среда разработки имеет ряд очень полезных функций, которые помогают разработчику повысить производительность при разработке программного обеспечения. Среди них:

– Очистка кода. Помогает решить проблемы в коде, прежде чем перейти к проверке кода;

– Рефакторинг. Включает такие операции, как интеллектуальное переименование переменных, извлечение одной или нескольких строк кода в новый метод, изменение порядка параметров метода и многое другое;

– Создание облачных приложений. Visual Studio предлагает набор инструментов, которые позволяют легко создавать облачные приложения на базе Microsoft Azure. Можно настраивать, создавать, отлаживать, упаковывать и развертывать приложения и службы в Microsoft Azure непосредственно из IDE;

– и др.

Интернет движет современным миром и Visual Studio помогает писать приложения для него. Данная среда разработки позволяет создавать веб-приложения, используя ASP.NET, Node.js, Python, JavaScript и TypeScript. Visual Studio понимает веб-фреймворки, такие как Angular, jQuery, Express и другие. ASP.NET Core и .NET Core работают в операционных системах Windows, Mac и Linux. ASP.NET Core представляет собой серьезное обновление для MVC, WebAPI и SignalR и работает на Windows, Mac и Linux.

### 1.3.2 Платформа .NET

Платформа – это «фундамент» приложения. От платформы зависит набор функций и возможностей разрабатываемого программного обеспечения [4].

.NET – это бесплатная кроссплатформенная платформа с открытым исходным кодом для разработки приложений различных типов.

Платформа .Net является универсальной и позволяет разработчику, пользуясь предоставленными возможностями, создавать разнообразные приложения, отвечающие всем требованиям рынка. В .NET можно использовать несколько языков, редакторов и библиотек для создания веб-приложений, игр и IoT.

Можно писать приложения .NET на C#, F# или Visual Basic.

– C# - простой, современный, объектно-ориентированный и типобезопасный язык программирования;

– F# — это кроссплатформенный функциональный язык программирования с открытым исходным кодом для .NET. Это также включает объектно-ориентированное и императивное программирование;

– Visual Basic — это доступный язык с простым синтаксисом для создания типобезопасных объектно-ориентированных приложений;

Для разработки веб-конвертера казахских текстов с кириллицы на латиницу, ключевым моментом является обработка содержимого файла. Данным требованиям полностью отвечает объектно-ориентированный язык программирования C# и поэтому именно он был выбран как основной язык.

Вне зависимости от того, какой язык программирования был использован, C#, F# или Visual Basic, код будет работать изначально на любой совместимой ОС. Различные реализации .NET справляются с тяжелой работой:

– .NET Core — это кроссплатформенная реализация .NET для веб-сайтов, серверов и консольных приложений в Windows, Linux и macOS;

– .NET Framework поддерживает веб-сайты, службы, настольные приложения и многое другое в Windows;

– Xamarin / Mono — это .NET-реализация для запуска приложений во всех основных мобильных операционных системах.

Из предоставленных возможностей, для решения поставленных в рамках данной дипломной работы задач, подходит фреймворк .NET Core.

Чтобы расширить функциональность, Microsoft и другие поддерживают здоровую экосистему пакетов, основанную на .NET Standard.

NuGet – это менеджер пакетов, созданный специально для .NET, который содержит более 90 000 пакетов.

### 1.3.3 Фреймворк ASP.NET

ASP.NET Core был разработан с нуля, чтобы предоставить разработчику компактный и компонуемый стек .NET для создания современных облачных веб-приложений и сервисов [5].

Производительность является ключевым направлением ASP.NET Core. Он быстрее, чем другие популярные веб-фреймворки в независимых тестах TechEmpower. И это очень важно, ведь одна из проблем, которую необходимо решить в рамках этой дипломной работы, является скорость обработки файла.

ASP.NET Core предназначен для быстрого развития компонентов, API-интерфейсов, компиляторов и языков, обеспечивая при этом стабильную и поддерживаемую платформу для поддержания работы приложений, что тоже является немаловажным преимуществом данного фреймворка [6].

Несколько версий ASP.NET Core могут существовать параллельно на одном сервере. Это означает, что одно приложение может принять последнюю версию, в то время как другие приложения продолжают работать в той версии, на которой они были протестированы [7].

ASP.NET Core предоставляет различные варианты поддержки жизненного цикла для удовлетворения потребностей веб-приложения.

Приложения ASP.NET можно разрабатывать и запускать в Windows, Linux, macOS и Docker. Это означает, что приложение, разработанное на этом фреймворке, без труда можно разместить на любом стандартном сервере, который поддерживает соответствующие операционные системы.

Поддержка и сервис необходимы большинству из разрабатываемые программных обеспечений. Microsoft выпускает официальные выпуски .NET, которые созданы и протестированы на поддерживаемых Microsoft серверах в Azure и поддерживаются, как и любой продукт Microsoft, что также является основной из причин, которые повлияли на выбор именно этого фреймворка в этой дипломной работе. Потому что как результат, необходимо веб-приложение, которое прослужит продолжительное время.

### 1.3.4 Язык программирования C#

C# (произносится «Си шарп») – это простой, современный, объектно-ориентированный и типобезопасный язык программирования [8].

C# называют языком программирования будущего, так как он не только отвечает всем современным требованиям, а еще развивается очень большим темпом. За счет своей универсальности и простоты стремительно занимает все больше и больше территории рынка программного обеспечения [9].

C # полагается на среду выполнения для автоматического управления памятью. Common Language Runtime имеет сборщик мусора, который выполняется как часть программы, освобождая память для объектов, на которые больше нет ссылок. Это освобождает программистов от явного освобождения памяти для объекта, устраняя проблему неправильных указателей, встречающихся в таких языках, как C++. C# не устраняет указатели: он просто делает их ненужными для большинства задач программирования. Для критических точек производительности и функциональной совместимости могут использоваться указатели, но они разрешены только в блоках, которые явно помечены как небезопасные. C# зависит от среды выполнения, оснащенной множеством функций, таких как автоматическое управление памятью и обработка исключений. Дизайн C# тесно связан с дизайном Microsoft Language Language Runtime (CLR), который обеспечивает эти функции времени выполнения (хотя C# технически не зависит от CLR). Кроме того, система типов C# отображается близко к системе типов CLR (например, оба используют одни и те же определения для предопределенных типов) [10].

### 1.3.5 Регулярные выражения

Язык регулярных выражений идентифицирует шаблоны символов. Типы .NET, поддерживающие регулярные выражения, основаны на

регулярных выражениях Perl 5 и поддерживают функции поиска и поиска / замены [11]. Регулярные выражения используются для таких задач, как:

– Проверка ввода текста, такого как пароли и номера телефонов (ASP.NET предоставляет элемент управления RegularExpressionValidator только для этой цели);

– Разбор текстовых данных в более структурированные формы (например, извлечение данных из HTML-страницы для хранения в базе данных);

– Замена шаблонов текста в документе (например, только целые слова)

Регулярные выражения – неотъемлемая часть программной части веб-конвертера, т.к. они используются при распознавании определенного набора знаков и по указанным правилам производит их замену на соответствующие знаки [12].

### 1.3.6 Шаблон проектирования MVC

MVC (обозначает «Модель-Вид-Контроллер».) – это модель разработки приложения, состоящая из трех взаимосвязанных частей. Они включают модель (данные), представление (пользовательский интерфейс) и контроллер (процессы, которые обрабатывают ввод) [13].

Модель MVC или «шаблон» обычно используются для разработки современных пользовательских интерфейсов. Это обеспечивает фундаментальные части для разработки программ для настольных или мобильных, а также веб-приложений. Он хорошо работает с объектно-ориентированным программированием, поскольку различные модели, представления и контроллеры можно рассматривать как объекты и повторно использовать в приложении. Именно поэтому его использование идеально вписывается в разрабатываемом в данном случае веб-конвертере.

### 1.3.7 Инструмент NuGet

Gакет NuGet представляет собой один ZIP-файл с расширением. nupkg, который содержит скомпилированный код (DLL), другие файлы, связанные с этим кодом, и описательный манифест, который включает в себя такую информацию, как номер версии пакета. Разработчики с кодом для совместного использования создают пакеты и публикуют их на общедоступном или частном хосте. Потребители пакетов получают эти пакеты от подходящих хостов, добавляют их в свои проекты и затем вызывают функциональность пакета в своем коде проекта. Затем NuGet обрабатывает все промежуточные детали.

Поскольку NuGet поддерживает частные хосты наряду с общедоступным хостом nuget.org, можно использовать пакеты NuGet для совместного использования кода, который является эксклюзивным для организации или рабочей группы. Также можно использовать пакеты NuGet в

качестве удобного способа разложить свой собственный код и использовать его только в своих проектах. Короче говоря, пакет NuGet является разделяемой единицей кода, но не требует и не подразумевает каких-либо конкретных средств совместного использования.

В рамках данной дипломной работы, использование инструмента NuGet обусловлено необходимостью внедрения пакета OpenXml.

### 1.3.8 Библиотека Open XML SDK

Open XML SDK предоставляет инструменты для работы с документами Office Word, Excel и PowerPoint [14]. Он поддерживает такие сценарии, как:

– Высокопроизводительное создание текстовых документов, электронных таблиц и презентаций;

– Заполнение содержимого в файлах Word из источника данных XML;

– Разделение (измельчение) файла Word или PowerPoint на несколько файлов и объединение нескольких файлов Word / PowerPoint в один файл;

– Извлечение данных из документов Excel;

– Поиск и замена содержимого в Word / PowerPoint с использованием регулярных выражений;

– Обновление кэшированных данных и встроенных электронных таблиц для диаграмм в Word / PowerPoint;

– Изменение документа, например добавление, обновление и удаление контента и метаданных;

SDK построен на API-интерфейсе System.IO.Packaging и предоставляет строго типизированные классы для управления документами, которые соответствуют спецификации форматов файлов Office Open XML. Спецификация форматов файлов Office Open XML является открытым международным стандартом ECMA-376, второго издания и ISO / IEC 29500. Форматы файлов Open XML полезны для разработчиков, поскольку они являются открытым стандартом и основаны на известных технологиях: ZIP и XML.

Данная библиотека – главный инструмент, который позволяет решить все поставленные задачи при разработке веб-конвертера казахских текстов с кириллицы на латиницу.

## 2 Специальная часть

### 2.1 Архитектура веб-приложения

Выбор модели и архитектуры может определить, насколько отзывчивым, надежным, безопасным и быстрым будет веб-приложение. Поэтому, прежде чем запускать проект разработки, необходим детальный анализ всех возможных вариантов.

Независимо от модели все компоненты веб-приложения всегда работают одновременно и создают единое веб-приложение. В зависимости от того, как логика приложения распределяется между клиентской и серверной сторонами, могут существовать различные типы архитектуры веб-приложений [15].

Архитектура одностраничного веб-приложения – это самая современная архитектура веб-приложений, где пользователь загружает одну страницу только один раз. На стороне клиента эта страница имеет слой JavaScript, который может свободно взаимодействовать с веб-службами на сервере и, используя данные веб-служб, обновлять себя в режиме реального времени. На рисунке 2.1, в виде диаграммы архитектуры веб-приложения, показано как это работает:

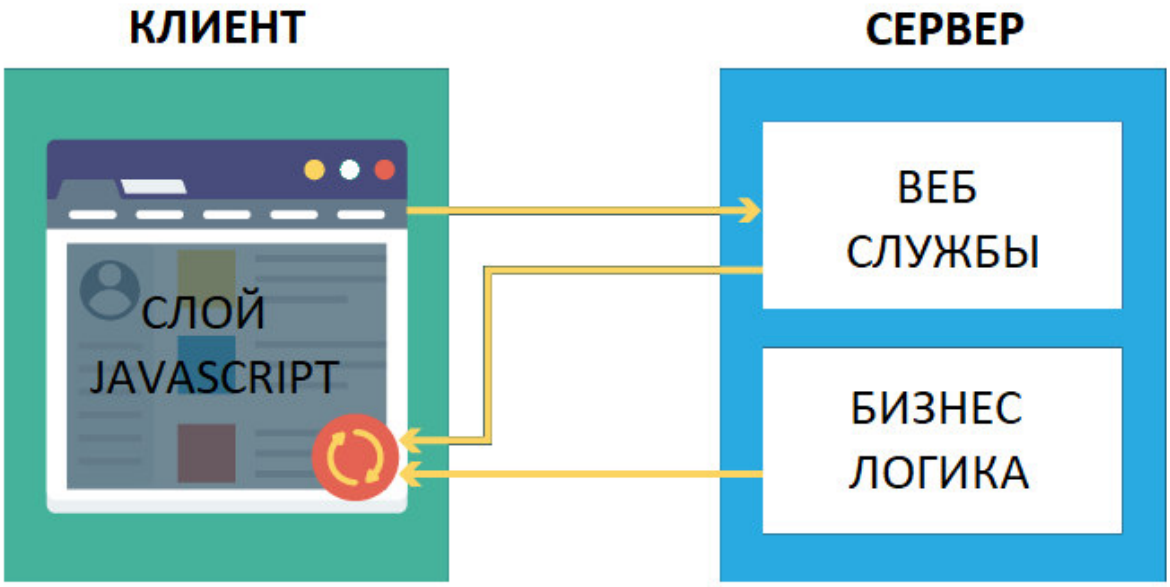


Рисунок 2.1 – Диаграмма архитектуры одностраничного веб-приложения

Куски данных, передаваемых с сервера на клиент, здесь минимальны, особенно по сравнению устаревшим типом, где пользователю приходилось каждый раз полностью перезагружать страницу, чтобы запрос отправился на сервер. Это очень гибкое, отзывчивое и легкое веб-приложение, которое можно легко администрировать и оптимизировать под любые нужды.

## 2.2 Структурные компоненты веб-конвертера

Двумя основными структурными компонентами веб-приложения являются клиентская и серверная части. Клиент — это удобное представление функциональности веб-приложения, с которым взаимодействует пользователь. Разработанный на HTML, JavaScript и CSS и существующий в веб-браузере пользователя, он не требует каких-либо особых настроек, связанных с операционной системой или устройством.

Для создания серверной части приложения использовался .NET. Эта сторона состоит как минимум из двух частей: логики приложения или главного центра управления и базы данных, где хранятся все постоянные данные. Есть и другие компоненты, которые тесно взаимосвязаны с серверной частью или являются ее дочерними компонентами.

## 2.3 Графический интерфейс пользователя

При создании графического интерфейса использовался универсальный инструмент для создания прототипов для веб-приложений Justinmind. Он позволяет создавать дизайн реалистичных прототипов, прорабатывать каждую деталь с помощью улучшенных взаимодействий, анимации и переходов. В нем можно добавлять динамический контент, показывать и скрывать элементы, ссылаться на другие экраны и многое другое.

Учитывая, что основной функционал приложения – конвертация казахских текстов, ожидается, что веб-конвертер будет находиться в эксплуатации на территории Республики Казахстан. Поэтому мульти-язычность – хороший способ расширить охват пользователей, для которых приложение будет полезным. Все страницы и элементы приложения представлены в двух языках: казахский и русский.

Главное окно веб-конвертера казахских текстов с кириллицы на латиницу представляет собой веб-страницу, которая разделена на секции. Эти секции можно условно разделить на:

– Заголовок;
– мини инструкция;
– главный функционал;
– дополнительный функционал;
– пояснение;
– дополнительные ссылки.


Рисунок 2.2 – Главная страница. Вариант на казахском языке


Рисунок 2.3 – Главная страница. Вариант на русском языке

Каждая из этих секций дополняют друг друга и не могут существовать самостоятельно.

Заголовок сразу дает посетителю веб-страницы понять, куда он попал и какие функции выполняет данное веб-приложение. Сразу под заголовком показаны три простых шага, которые посетитель должен сделать, чтобы воспользоваться функционалом приложения. А ниже ему предоставляется доступ к самым главным функциям – конвертации казахского текста. Сразу после секций с функционалом размещена таблица, в которой отображается казахский алфавит, основанный на латинской графике с соответствующими буквами алфавита на кириллице.

В целом, поведенческий фактор отражает напрямую на сколько удобно пользоваться веб-приложением. В совокупности влияют факторы структуры страницы, графической составляющей и контента страницы. Именно такая последовательность дает пользователю понимать общую картину веб-приложения и понимать его назначение. Такое правило построения страницы положительно влияет на поведенческий фактор.

Еще одной страницей приложения является «Инструкция». На ней пользователь может узнать, как преобразовать файл к формату, который поддерживает приложение. Данная страница также представлена в двух языковых вариантах: казахский и русский.

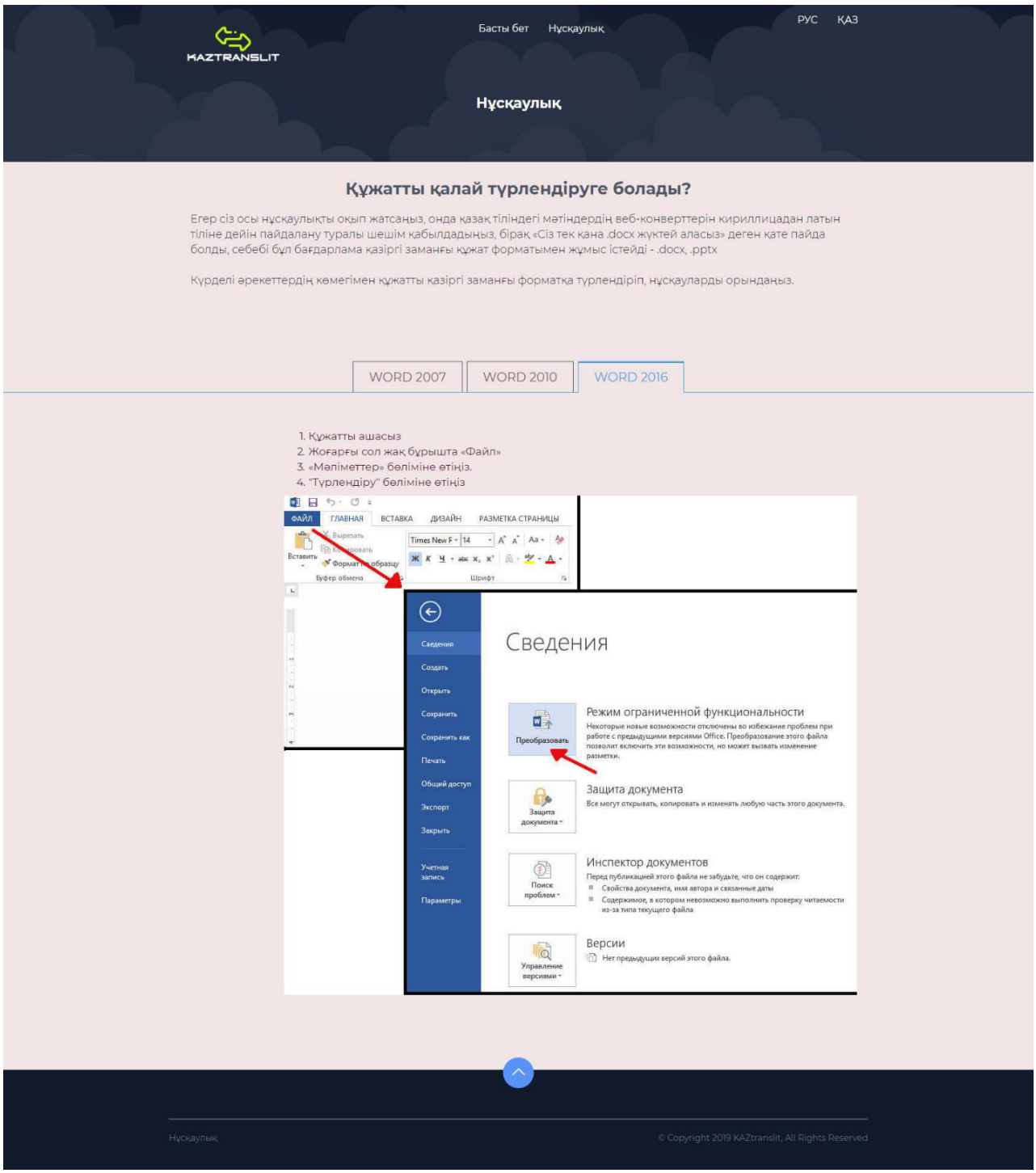


Рисунок 2.4 – Страница «Инструкция». Вариант на казахском языке

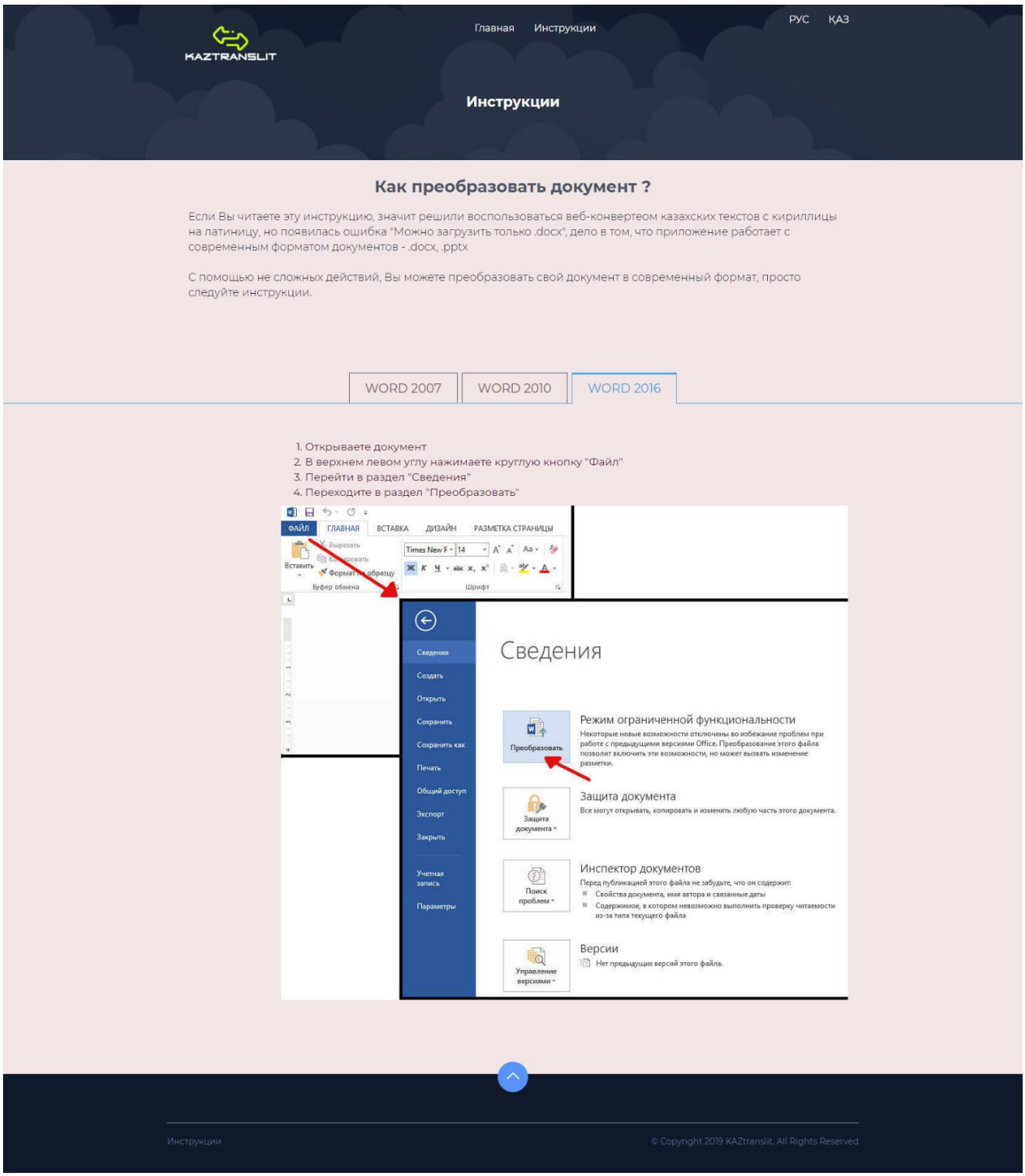


Рисунок 2.5 – Страница «Инструкция». Вариант на русском языке

На этапе создания интерфейса были взяты во внимание и реализованы основные принципы проектирования пользовательского интерфейса [16].

Среди таких принципов можно выделить наиболее важные, такие как: смешанный минимализм, выразительная типография, яркие и смелые оттенки.

Минимализм был одной из самых популярных тенденций веб-дизайна с начала 2000-х годов. Он направлен на уменьшение когнитивной перегрузки и повышение удобства использования пользовательских интерфейсов, подчеркивая простоту и удаляя лишние элементы. Минималистские дизайны пользовательского интерфейса обычно имеют большое количество пустого пространства, чистые и четкие края и ограниченную цветовую палитру, чтобы привлечь внимание пользователя к важным элементам.

Типография всегда была важной частью дизайна пользовательского интерфейса, помогая сосредоточить внимание пользователя, вызвать эмоции и задать тон контента. Благодаря чистым и простым макетам, в которых доминируют высочайшие и насыщенные изображениями конструкции, занимающие заднее сиденье, мы видим смелые, яркие шрифты в центре внимания.

Цвет оказывает огромное влияние на то, как пользователи воспринимают информацию, влияя как на их эмоции, так и на коэффициент конверсии. В интерфейсе используется сочетание синего, голубого и желтого цветов.

### 2.4 Программная часть веб-приложения

Все этапы разработки и отладки программной части веб-конвертера были выполнены в среде разработки Visual Studio.

Разработка приложения начинается с создания проекта в среде разработки. В логическом смысле проект содержит все файлы исходного кода, значки, изображения, файлы данных и т.д., которые скомпилированы в исполняемый файл, библиотеку или веб-сайт. Проект также содержит настройки компилятора и другие файлы конфигурации, которые могут понадобиться различным службам или компонентам, с которыми взаимодействует веб-приложение.

В Visual Studio файл проекта используется обозревателем решений для отображения содержимого и настроек проекта. Когда происходит компиляция проекта, движок MSBuild использует файл проекта для создания исполняемого файла. Кроме того, можно настроить проекты для получения других видов вывода.

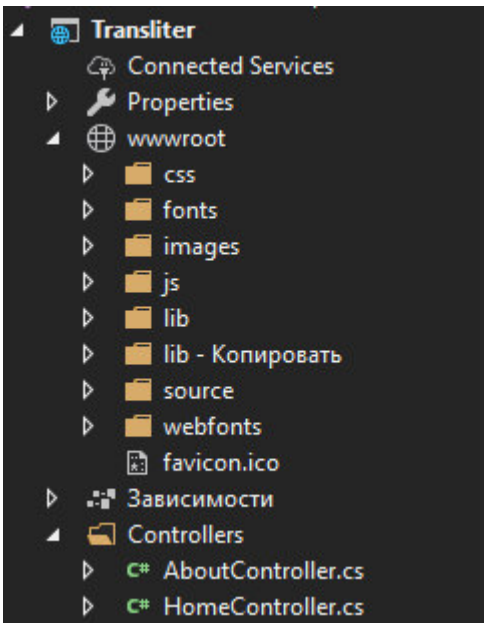


Рисунок 2.6 – Дерево проекта. Часть 1

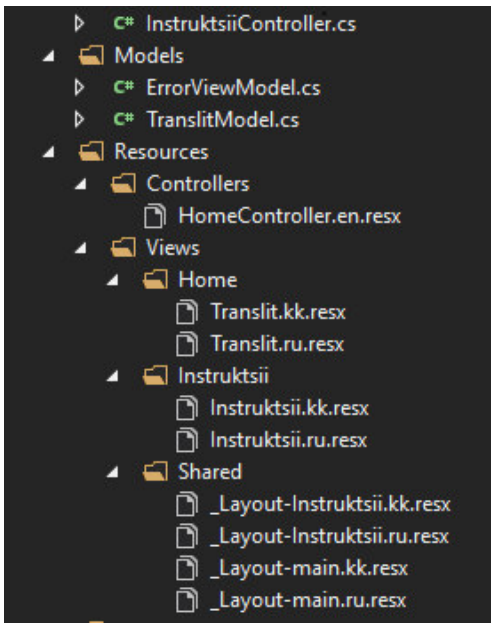


Рисунок 2.7 – Дерево проекта. Часть 2

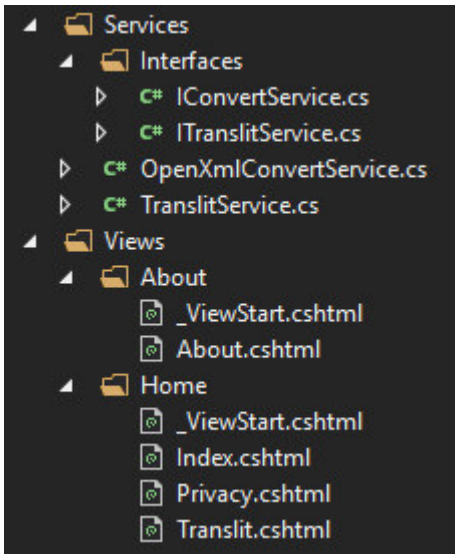


Рисунок 2.8 – Дерево проекта. Часть 3

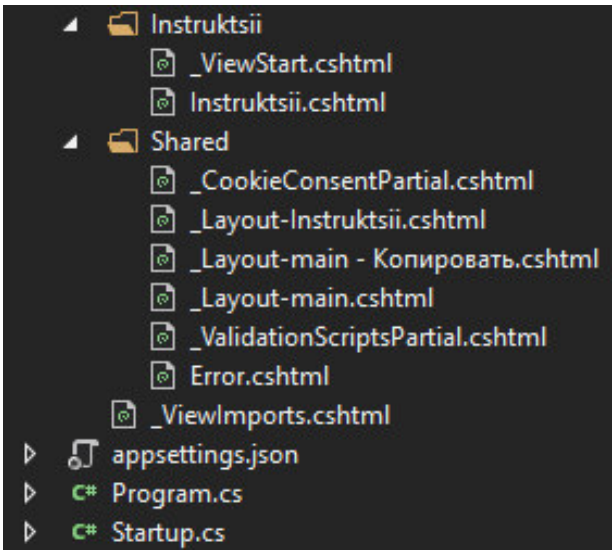


Рисунок 2.9 – Дерево проекта. Часть 4

На рисунках 2.6, 2.7, 2.8 и 2.9 показано дерево проекта «веб-конвертер казахских текстов с кириллицы на латиницу».

Данное дерево содержит все элементы, использованные при построении модели MVC.

Model-View-Controller (MVC) – это архитектурный шаблон, который разделяет приложение на три основных логических компонента: модель, представление и контроллер (см. рисунок 2.10). Каждый из этих компонентов создан для обработки определенных аспектов разработки приложения. MVC является одной из наиболее часто используемых отраслевых сред разработки веб-приложений для создания масштабируемых и расширяемых проектов.


Рисунок 2.10 – Диаграмма MVC

### 2.5 Логический компонент «Модель»

Компонент «Модель» соответствует всей логике, связанной с данными, с которыми работает пользователь. Это могут быть либо данные, которые передаются между компонентами «Представление» и «Контроллер», либо любые другие данные, связанные с бизнес-логикой.

В рамках разработки веб-конвертера казахских текстов с кириллицы на латиницу предполагается, что данными, с которыми работает пользователь, является документ, который он загружает на сервер и который подлежит дальнейшей латинизации его содержимого.

Файлы, которые составляют логический компонент «Модель» в структуре данного приложения располагаются в соответствующей папке «Models». Получение данных реализовано в файле TranslitModel.cs (см. рисунок 2.11).

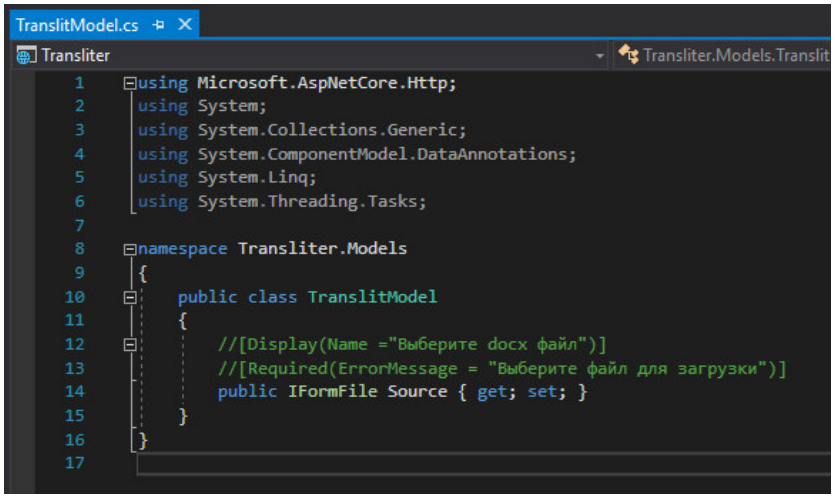


Рисунок 2.11 – Метод TranslitModel

Метод TranslitModel получает данные формы, у которой установлен признак Source в виде документа, который был загружен пользователем, и передает его в другой логический компонент – «Контроллер».

### 2.6 Логический компонент «Представление»

Компонент «Представление» используется для всей логики пользовательского интерфейса приложения, с которыми взаимодействует конечный пользователь. В нашем случае это страница, которую он видит, посетив веб-ресурс.

В дереве проекта файлы логического компонента располагаются в папке «Views». В этой папке разные представления отделяются друг от друга дополнительными папками с соответствующими названиями (см. рисунок 2.8).

Веб-конвертер состоит из двух страниц, значит и представлений тоже два.

Представление главной страницы (в рамках данной дипломной работы оно называется Translit) имеет свою разметку страницы. Файл разметки страницы Translit находится в папке Shared, которая является дочерней папкой Views, и называется _Layout-main.cshtml (см. рисунок 2.9).

Для исполнения данного файла метод IActionResult Translit() обращается к файлу _ViewStart.cshtml, который обязательно должен находиться в той же папке, где находится само представление Translit (см. рисунок2.8).

Файл ViewStart.cshtml содержит команду (см. рисунок 2.12).

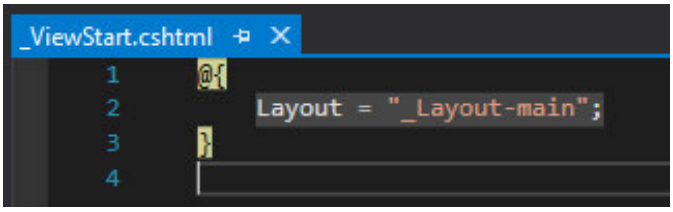


Рисунок 2.12 – Файл ViewStart.cshtml

Данная конструкция указывает на файл, в котором находится разметка для представления. В данном случае программа переходит в файл _Layout-main.cshtml. Этот файл содержит HTML код разметки, который создает основную DOM структуру.

В результате, пользователь веб-конвертера видит на экране главную страницу приложения, с которой и будет взаимодействовать на протяжении всего времени эксплуатации веб-конвертера.

Главная страница – есть представление, которое является совокупностью всех объектов и элементов приложения, с которыми непосредственно взаимодействует пользователь.

Вторым представлением приложения является страница «Инструкции». За эту страницу отвечает файл представления Instruktsii.cshtml и

соответствующий ему файл разметки _ViewStart.cshtml, который также находится с ним в одной папке (см. рисунок 2.9)

Файл ViewStart.cshtml содержит:

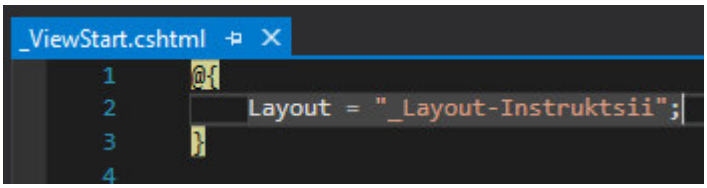


Рисунок 2.13 – Файл ViewStart.cshtml

### 2.7 Логический компонент «Контроллер»

«Контроллеры» приложения в дереве проекта находятся в соответствующей папке «Controllers» (см. рисунок 2.6). Для каждого отдельного представления создается свой контроллер.

Контроллеры данного веб-приложения: HomeController.cs и InstruktsiiController.cs.

HomeController.cs – основной контроллер приложения, который относится к главной странице веб-конвертера и содержит в себе методы основного функционала.

В начале класса HomeController объявляются основные сервисы приложения (см. рисунок 2.14).

```
private readonly IConvertService convertService;
private readonly IHostingEnvironment env;

public HomeController(IConvertService convertService, IHostingEnvironment env)
{
    this.convertService = convertService;
    this.env = env;
}
```

Рисунок 2.14 – Сервисы приложения

Метод, который указывает какое представление следует вызывать:

```
public IActionResult Translit()
{
    return View();
}
```

Рисунок 2.15 – Метод Translit

Метод Translit (см. рисунок 2.15), который получает данные из файла модели TranslitModel.cs. Он начинается с условия «if (ModelState.IsValid)».

Оно проверяет, были ли получены данные для обработки. В случае, если условие удовлетворяется, происходит запуск алгоритма.

Полученные данные необходимо объявить в программе (см. рисунок 2.16).

```
var fileName = model.Source.FileName;
```

Рисунок 2.16 - Объявление переменной для файла

Так как веб-конвертер рассчитан на работу с определенными форматами документов, необходима проверка формата. Получаем формат документа (см. рисунок 2.17).

```
var ext = Path.GetExtension(fileName);
```

Рисунок 2.17 – Переменная для расширения файла

Загруженный на сервер файл необходимо как-то идентифицировать, поэтому присваиваем ему имя (см. рисунок 2.18).

```
var fn = Guid.NewGuid() + ext;
```

Рисунок 2.18 – Переменная для нового имени файла

Для дальнейшей работы с данным файлом необходимы данные о его размещении на сервере (см. рисунок 2.19).

```
var path = Path.Combine(env.WebRootPath, "source", fn);
```

Рисунок 2.19 – Размещение файла на сервере

Сохраняем подготовленный для обработки файл (см. рисунок 2.20).

```
using (var stream = new FileStream(path, FileMode.CreateNew))
{
    await model.Source.CopyToAsync(stream);
}
```

Рисунок 2.20

На данном этапе получена вся информация о данных, пришедших из модели. Это делает возможным проверить, на сколько возможно их обрабатывать. Проводим проверку расширения файла (см. рисунок 2.21).

```
if (ext == ".docx")
{
    convertService.ConvertDOC(path);
    //возвращаем обратно измененный
    return PhysicalFile(path,
        "application/vnd.openxmlformats-officedocument.wordprocessingml.document",
        fileName);
}
else if (ext == ".pptx")
{
    convertService.ConvertPPT(path);
    //возвращаем обратно измененный
    return PhysicalFile(path,
        "application/vnd.openxmlformats-officedocument.presentationml.presentation",
        fileName);
}
else
{
    ModelState.AddModelError("Source", "Можно загрузить только docx файлы.");
    return View(model);
}
```

Рисунок 2.21 – Проверка расширения файла

Данная конструкция состоит из трех условий: если расширение файла «.docx», если расширение файла «.pptx», если другое расширение.

В случае, если расширение файла не равно «.docx» или «.pptx» в представлении отображается текст ошибки «Можно загрузить только .docx/.pptx файлы». Таким образом пользователю становится известно, что нужно загрузить файл с другим расширением.

Если расширение файла равно «.docx», запускается метод convertService.ConvertDOC, после чего пользователю возвращается файл для загрузки (return PhysicalFile). Если расширение равно «.docx», запускается метод convertService. ConvertPPT, после чего пользователю также возвращается файл для загрузки (return PhysicalFile).

Оба метода ConvertDOC и ConvertPPT прописаны в сервисе IConvertService.cs (см. рисунок 2.8). А ход исполнения данных методов представлен в файле OpenXmlConvertService.cs.

Функция ConvertDOC с входным параметром расположения документа на сервере содержит следующий код:

```
public void ConvertDOC(string path)
{
    //открываем docx
    using (var doc = WordprocessingDocument.Open(path, true))
    {
        //получаем тело
        var body = doc.MainDocumentPart.Document.Body;

        //для всех входящих элементов
        foreach (var text in body.Descendants<Text>())
        {
            //translit
            text.Text = translitService.Convert(text.Text);
        }
        //сохраняем изменения
        doc.Save();
    }
```

Рисунок 2.22 – Функция ConvertDOC

В теле этого метода (см. рисунок 2.22) вызывается функция Convert, которая находится в сервисах приложения. ITranslitService объявляет словарь (см. рисунок 2.23).

```
private static Dictionary<string, string> tdict = new Dictionary<string, string>();
```

Рисунок 2.23 – Объявление словаря

В словаре определяются значения букв алфавита и знаков (см. рисунки 2.24, 2.25).

```
static TranslitService()
{
    //определяем латинский казахский алфавит
    tdict.Add("А", "A"); tdict.Add("а", "a");
    tdict.Add("Ә", "Á");    tdict.Add("ә", "á");
    tdict.Add("Б", "B");    tdict.Add("б", "b");
    tdict.Add("В", "V");    tdict.Add("в", "v");
    tdict.Add("Г", "G");    tdict.Add("г", "g");
    tdict.Add("Ғ", "Ǵ");    tdict.Add("ғ", "ǵ");
    tdict.Add("Д", "D");    tdict.Add("д", "d");
    tdict.Add("Е", "E");    tdict.Add("е", "e");
    tdict.Add("Ё", "Ë");    tdict.Add("ё", "ë");
    tdict.Add("Ж", "J");    tdict.Add("ж", "j");
    tdict.Add("З", "Z");    tdict.Add("з", "z");
    tdict.Add("И", "I");    tdict.Add("и", "ı");
    tdict.Add("Й", "I");    tdict.Add("й", "ı");
    tdict.Add("К", "K");    tdict.Add("к", "k");
    tdict.Add("Қ", "Q");    tdict.Add("қ", "q");
    tdict.Add("Л", "L");    tdict.Add("л", "l");
    tdict.Add("М", "M");    tdict.Add("м", "m");
    tdict.Add("Н", "N");    tdict.Add("н", "n");
    tdict.Add("Ң", "Ń");    tdict.Add("ң", "ń");
    tdict.Add("О", "O");    tdict.Add("о", "o");
    tdict.Add("Ө", "Ó");    tdict.Add("ө", "ó");
    tdict.Add("П", "P");    tdict.Add("п", "p");
    tdict.Add("Р", "R");    tdict.Add("р", "r");
    tdict.Add("С", "S");    tdict.Add("с", "s");
    tdict.Add("Т", "T");    tdict.Add("т", "t");
    tdict.Add("У", "Ý");    tdict.Add("у", "ý");
    tdict.Add("Ұ", "U");    tdict.Add("ұ", "u");
    tdict.Add("Ү", "Ú");    tdict.Add("ү", "ú");
    tdict.Add("Ф", "F");    tdict.Add("ф", "f");
    tdict.Add("Х", "H");    tdict.Add("х", "h");
    tdict.Add("Һ", "H");    tdict.Add("һ", "h");
```

Рисунок 2.24 – Значения словаря (часть 1)

```
    tdict.Add("Ц", "Ц");    tdict.Add("ц", "ц");
    tdict.Add("Ч", "Ch");   tdict.Add("ч", "ch");
    tdict.Add("Ш", "Sh");   tdict.Add("ш", "sh");
    tdict.Add("Щ", "Щ");    tdict.Add("щ", "щ");
    tdict.Add("Ъ", "Ъ");    tdict.Add("ъ", "ъ");
    tdict.Add("Ы", "Y");    tdict.Add("ы", "y");
    tdict.Add("I", "I");    tdict.Add("i", "i");
    tdict.Add("Ь", "Ь");    tdict.Add("ь", "ь");
    tdict.Add("Э", "Э");    tdict.Add("э", "э");
    tdict.Add("Ю", "Iу́");  tdict.Add("ю", "ıу́");
    tdict.Add("Я", "Ia");   tdict.Add("я", "ıa");



    //определяем знаки
    tdict.Add(" ", " ");    tdict.Add("«", "«");
    tdict.Add("!", "!");    tdict.Add("»", "»");
    tdict.Add("@", "@");    tdict.Add("{", "{");
    tdict.Add("#", "#");    tdict.Add("}", "}");
    tdict.Add("$", "$");    tdict.Add("[", "[");
    tdict.Add("%", "%");    tdict.Add("]", "]");
    tdict.Add("^", "^");    tdict.Add(",", ",");
    tdict.Add("&", "&");    tdict.Add(".", ".");
    tdict.Add("*", "*");    tdict.Add("№", "№");
    tdict.Add("(", "(");    tdict.Add(";", ";");
    tdict.Add(")", ")");    tdict.Add(":", ":");
    tdict.Add("_", "_");    tdict.Add("?", "?");
    tdict.Add("+", "+");    tdict.Add("<", "<");
    tdict.Add("=", "=");    tdict.Add(">", ">");
    tdict.Add("-", "-");
}
```

Рисунок 2.25 - Значения словаря (часть 2)

Текст проходит обработку регулярными выражениями (см. рисунок 2.26).

```
output = Regex.Replace(output, @"\s|\(", " ");
output = Regex.Replace(output, @"\s+", " ");
output = Regex.Replace(output, @"[^\S\W\D-]", "");
```

Рисунок 2.26 – Регулярные выражения

В этом коде:

- \s - Соответствует любому знаку пробела;
- \S - Соответствует любому знаку, не являющемуся пробелом;
- \W - Соответствует любому символу, который не является буквенно-цифровым знаком.
- \D - Соответствует любому символу, не являющемуся десятичной цифрой

Результат возвращается с помощью выражения «return output;» в OpenXmlConvertService.cs.

Функция ConvertPPTc входным параметром расположения документа на сервере содержит следующий код:

```
public void ConvertPPT(string path)
{
    int numberOfSlides = CountSlides(path);
    //string slideText;
    for (int i = 0; i < numberOfSlides; i++)
    {
        GetSlideIdAndText(path, i);
    }

}
```

Рисунок 2.27 - Функция ConvertPPT

Как видно из представленного в теле ConvertPPT вызывается последовательно две функции: CountSlides (см. рисунок 2.28) и GetSlideIdAndText.

Функция CountSlides содержит:

```
public static int CountSlides(string path)
{
    // Open the presentation as read-only.
    using (PresentationDocument doc = PresentationDocument.Open(path, false))
    {
        // Pass the presentation to the next CountSlides method
        // and return the slide count.
        return CountSlides(doc);
    }
}
```

Рисунок 2.28 – Метод CountSlides

В своем теле она вызывает функцию CountSlides:

```
public static int CountSlides(PresentationDocument doc)
{
    // Check for a null document object.
    if (doc == null)
    {
        throw new ArgumentNullException("presentationDocument");
    }

    int slidesCount = 0;

    // Get the presentation part of document.
    PresentationPart presentationPart = doc.PresentationPart;
    // Get the slide count from the SlideParts.
    if (presentationPart != null)
    {
        slidesCount = presentationPart.SlideParts.Count();
    }
    // Return the slide count to the previous method.
    return slidesCount;
}
```

Рисунок 2.29 – Функция CountSlides

Проверяется пустой файл или нет и если не пустой, то возвращается количество слайдов в презентации.

После чего программа переходит к методу GetSlideIdAndText, который делает транслитирацию с помощью описанной выше функции translitService.Convert (см. рисунок 2.30).

```
public void GetSlideIdAndText(string path, int index)
{
    using (PresentationDocument ppt = PresentationDocument.Open(path, false))
    {
        // Get the relationship ID of the first slide.
        PresentationPart part = ppt.PresentationPart;
        OpenXmlElementList slideIds = part.Presentation.SlideIdList.ChildElements;

        string relId = (slideIds[index] as SlideId).RelationshipId;

        // Get the slide part from the relationship ID.
        SlidePart slide = (SlidePart)part.GetPartById(relId);

        // Build a StringBuilder object.
        StringBuilder paragraphText = new StringBuilder();

        // Get the inner text of the slide:
        IEnumerable<A.Text> texts = slide.Slide.Descendants<A.Text>();
        foreach (A.Text text in texts)
        {
            text.Text = translitService.Convert(text.Text);
        }
        ppt.Save();
    }
}
```

Рисунок 2.30 – Функция translitService.Convert

Контроллером для страницы «Инструкции» выступает файл InstruktsiiController.cs. В начале этого конструктора также инициализируются сервисы (см. рисунок 2.31):

```
private readonly IHostingEnvironment env;
```

Рисунок 2.31 – Сервис

А также метод, возвращающий представление «Instruktsii.cshtml»:

```
public IActionResult Instruktsii()
{
    return View();
}
```

Рисунок 2.32 – Метод Instruktsii

### 2.8 Запуск веб-приложения

В запуске приложения главную роль играет файл Startup.cs (см. рисунок 2.9). В этом файле реализованы методы Configure() (см. рисунок 2.34), который используется для добавления сервисов в контейнер, и ConfigureServices() (см. рисунок 2.33), который используется для настройки конвейера HTTP-запроса. Оба этих метода вызываются во время выполнения.

```
// Этот метод вызывается во время выполнения. Используется
//для добавления сервисов в контейнер
public void ConfigureServices(IServiceCollection services)
{
    services.AddLocalization(options => options.ResourcesPath = "Resources");
    services.AddMvc()
      .AddViewLocalization(LanguageViewLocationExpanderFormat.Suffix)
      .AddDataAnnotationsLocalization();

    services.Configure<CookiePolicyOptions>(options =>
    {
        // Эта лямбда-функция определяет, требуется ли согласие пользователя
        //для несущественных файлов cookie для данного запроса.
        options.CheckConsentNeeded = context => true;
        options.MinimumSameSitePolicy = SameSiteMode.None;
    });


    services.AddTransient<IConvertService, OpenXmlConvertService>();
    services.AddTransient<ITranslitService, TranslitService>();

    services.AddMvc().SetCompatibilityVersion(CompatibilityVersion.Version_2_2);
}
```

Рисунок 2.33 – Метод ConfigureServices

```
public void Configure(IApplicationBuilder app, IHostingEnvironment env)
{
    if (env.IsDevelopment())
    {
        app.UseDeveloperExceptionPage();
    }
    else
    {
        app.UseExceptionHandler("/Home/Error");
    }

    var supportedCultures = new[]
    {
        new CultureInfo("ru-RU"),
        new CultureInfo("kk-KZ"),

    };
    app.UseRequestLocalization(new RequestLocalizationOptions
    {
        DefaultRequestCulture = new RequestCulture("ru-RU"),
        SupportedCultures = supportedCultures,
        SupportedUICultures = supportedCultures
    });

    app.UseStaticFiles();
    app.UseCookiePolicy();

    app.UseMvc(routes =>
    {
        routes.MapRoute(
            name: "default",
            template: "{culture=en}/{controller=Home}/{action=Translit}/{id?}");
    });
}
```

Рисунок 2.34 – Метод Configure

В свою очередь метод Configure определяет, что исполняющим файлом объявляется контроллер HomeController.cs (см. рисунок 2.6) и класс Translit.cshtml (см. рисунок 2.8). Запуск исполняющего файла и сопутствующих ему методов и функций занимает 323.6081 миллисекунды (см. рисунок 2.35).

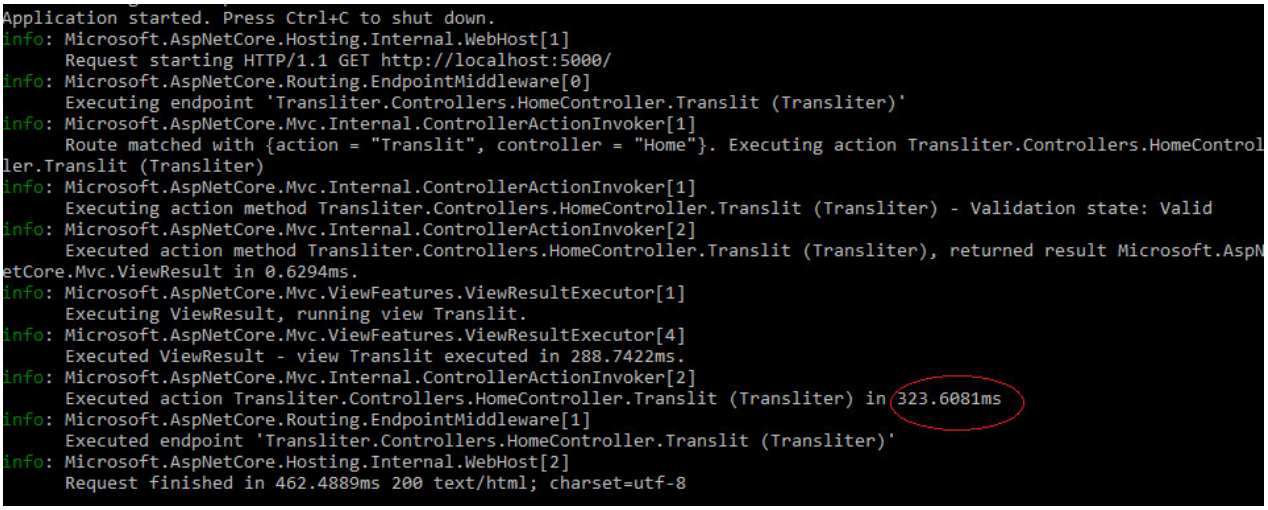

Рисунок 2.35 – Консоль работы приложения

### 2.9 Библиотека «DocumentFormat.OpenXml»

Для того чтобы добавить библиотеку DocumentFormat.OpenXml необходимо запустить Developer Command Prompt for VS и перейти к расположению библиотеки в хранилище компьютера (см. рисунок 2.36).

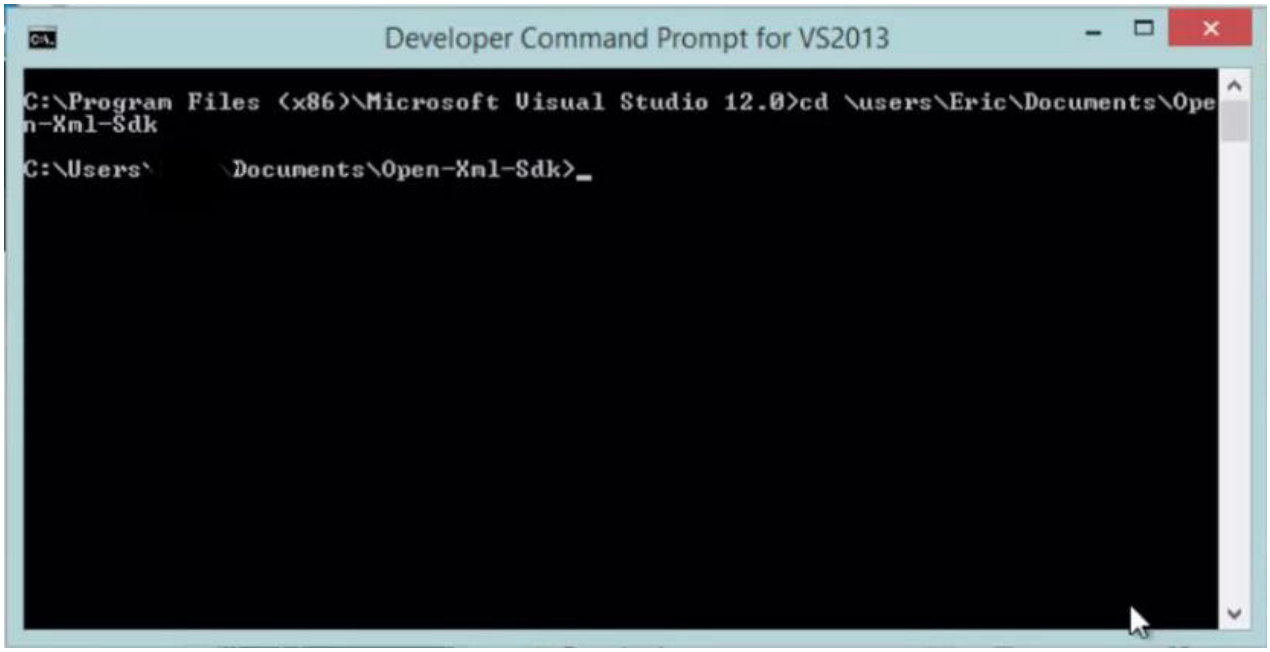

Рисунок 2.36 – Developer Command Prompt for VS

После чего ввести команду msbuild Open-Xml-SDK-Orig-SIP.sln (см. рисунок 2.37).

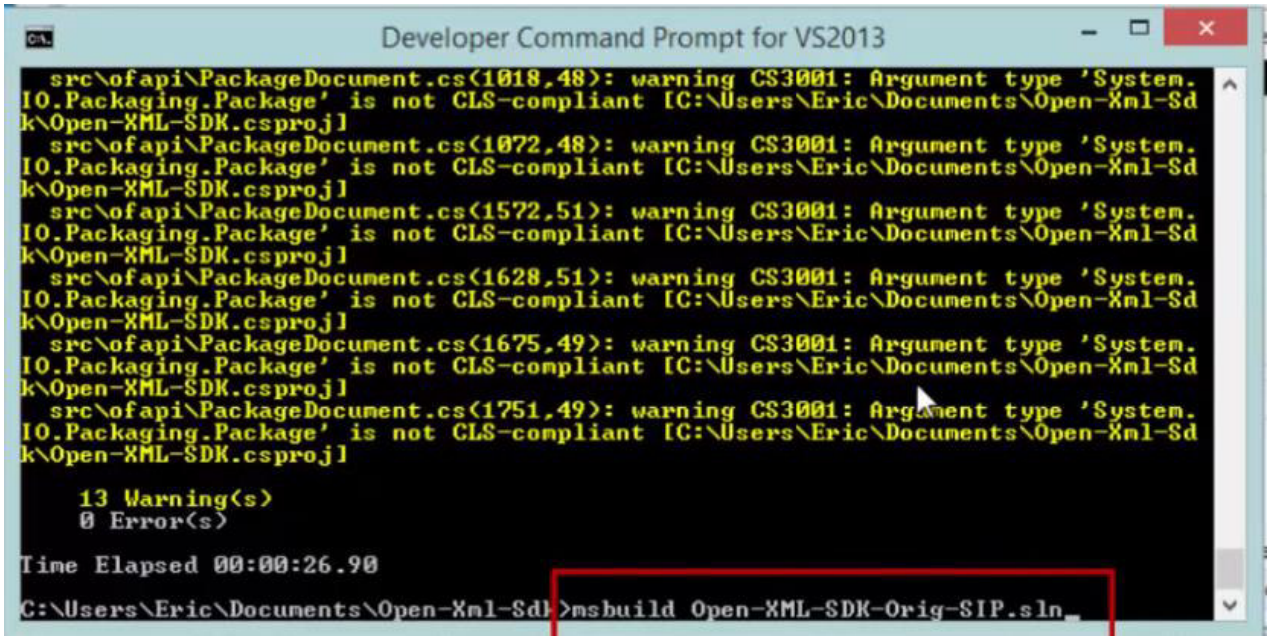


Рисунок 2.37 – Команда установки в консоли

В результате этих действий была установлена библиотека. Но ее осталось добавить в проект. Чтобы это сделать необходимо зайти в Диспетчер пакетов NuGet (см. рисунок 2.38).

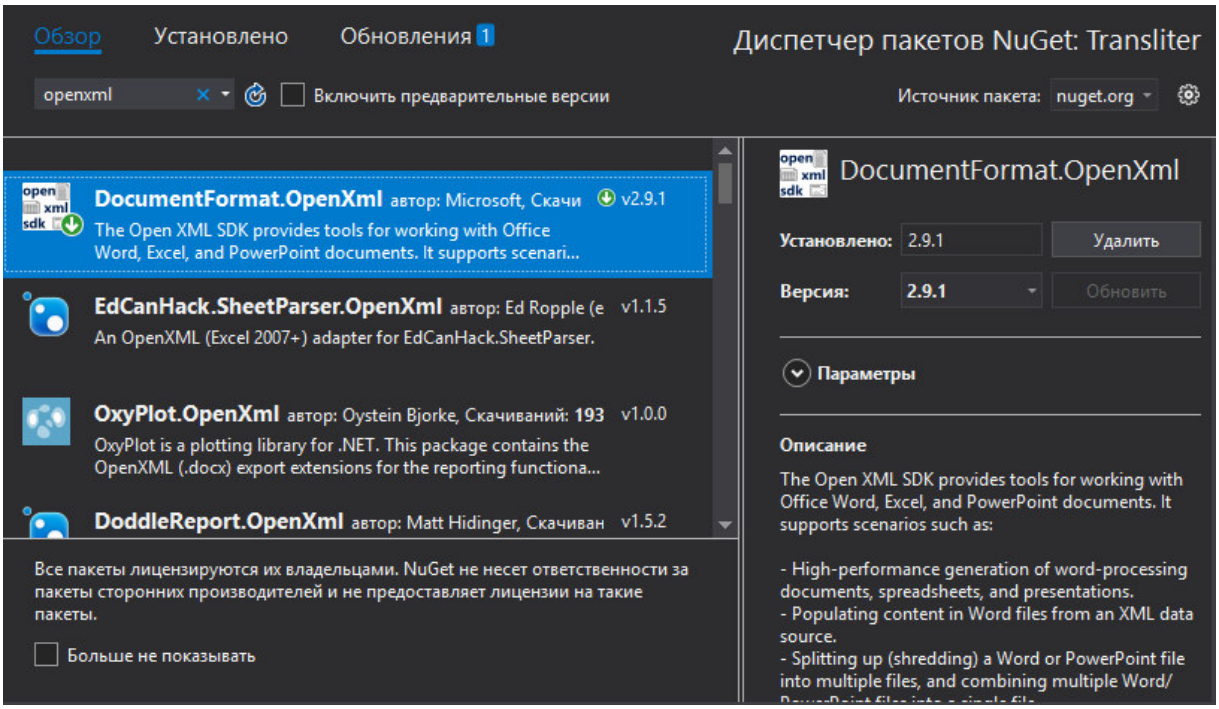


Рисунок 2.38 – Диспетчер пакетов NuGet

В дереве проекта эта библиотека находится в папке «Зависимости-NuGet».

### 2.10 Локализация и глобализация веб-конвертера

ASP.NET Core предоставляет сервисы и промежуточное программное обеспечение для локализации на разные языки и культуры.

Интернационализация включает в себя глобализацию и локализацию. Глобализация – это процесс разработки приложений, которые поддерживают разные культуры. Глобализация добавляет поддержку ввода, отображения и вывода определенного набора языковых сценариев, которые относятся к конкретным географическим областям.

Локализация приложения включает в себя следующее:

– Сделать контент приложения локализуемым;

– предоставить локализованные ресурсы для языков и культур, которые будут поддерживаться;

– реализовать стратегию, чтобы выбрать язык / культуру для каждого запроса.

Использование ресурсов – это быстрый и простой способ локализации веб-ресурса. Файлы локализации представляют собой файлы ресурсов с расширением .resx. Они все располагаются в одной папке «Resources» (см. рисунок 2.7). При необходимости локализации нескольких типов файлов, они разделяются на соответствующие папки внутри папки «Resources».

Как пример рассмотрим файл ресурса Translit.kk.resx. Название до точки соответствует названию представления, к которому относится данный файл ресурса. Двойной символ после точки соответствует коду языка, на котором представлена информации в представлении. Далее в названии указывается расширение файла.

Файл ресурса представляет собой таблицу значений с тремя колонками: Имя, Значение, Комментарий (см. рисунок 2.39).

| Имя | Значение | Комментарий |
|---|---|---|
| Column pronounce | Айтылуы | |
| Column write | Жазылуы | |
| Initial text | Бастапқы мәтін | |
| Submit | Жіберу | |
| Table-title | Қазақ тіліндегі латын алфавиті | |
| Text translit description | Тым көп мәтін болмаса, тез және ыңғайлы. | |
| Text translit title | Мәтін өрісін аудару үшін пайдалануға болады. | |
| Translate button | Аудару | |
| Translated text | Аударылған мәтін | |
| Transliter description text | Microsoft Office құжат нұсқасының 2007 нұсқасын немесе одан жаңасын таңдаңыз: Word (docx), PowerPoint (pptx) | |
| Upload button text | Файлды жүктеу | |
| | | |

Рисунок 2.39 – Файл ресурса Translit.kk.resx

Имя – это код значения, который указывается в файле представления

Значение – это текст, который соответствует имени, которое будет показано в итоге пользователю [17].

Комментарий – это информация, которая не доступна пользователю, а служит только как пояснение для разработчика или администратора веб-приложения.

Для того чтобы компилятор понимал данную конструкцию необходимо добавить сервисы в главные методы приложения ConfigureServices и Configure (см. рисунок 2.40).

В ConfigureServices добавляем:

```
services.AddLocalization(options => options.ResourcesPath = "Resources");
services.AddMvc()
  .AddViewLocalization(LanguageViewLocationExpanderFormat.Suffix)
  .AddDataAnnotationsLocalization();
```

Рисунок 2.40 – Сервисы

В Configure добавляем:

```
if (env.IsDevelopment())
{
    app.UseDeveloperExceptionPage();
}
else
{
    app.UseExceptionHandler("/Home/Error");
}

var supportedCultures = new[]
{
    new CultureInfo("ru-RU"),
    new CultureInfo("kk-KZ"),

};
```

Рисунок 2.41 – Добавление языков

Указываем язык по умолчанию (см. рисунок 2.42).

```
app.UseRequestLocalization(new RequestLocalizationOptions
{
    DefaultRequestCulture = new RequestCulture("ru-RU"),
    SupportedCultures = supportedCultures,
    SupportedUICultures = supportedCultures
});
```

Рисунок 2.42 – Установка языка по умолчанию

Также необходимо инициализировать все эти сервисы. Для этого в самом верху содержимого файла Startup.cs добавляем (см. рисунок 2.43):

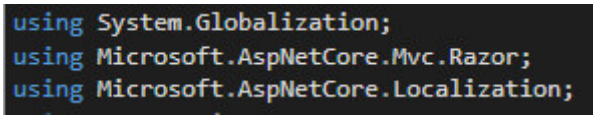


Рисунок 2.43 – Инициализация Сервисов

Конструкция для использования в представлениях, контроллерах или файлах разметки представлений (см. рисунок 2.44).

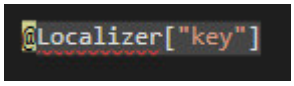


Рисунок 2.44 – Ключевое слово

Для того чтобы сделать эту конструкцию доступной, в нужном файле вверху необходимо добавить строки (см. рисунок 2.45):

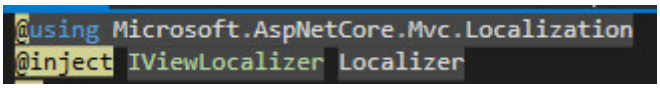


Рисунок 2.45 – Сервисы

Пользователю предоставлена возможность удобно переключаться между языками (см. рисунок 2.46).

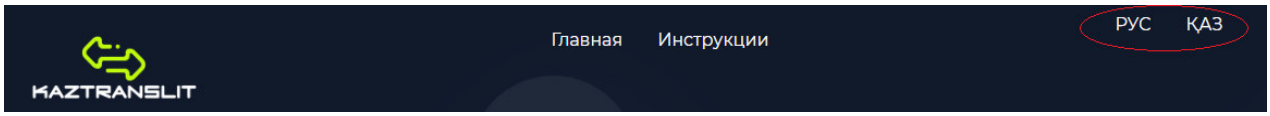


Рисунок 2.46 – Верхнее меню веб-приложения

## 3 Экономическое обоснование разработки проекта

Тема дипломного проекта – «Разработка веб-конвертера казахских текстов с кириллицы на латиницу».

Целью экономического обоснования дипломного проекта является расчет и анализ затрат, необходимых для создания и реализации веб-конвертера, изучить затраты на этапах проектирования, разработки, внедрения и функционирования приложения; определить условия и сроки окупаемости затрат. Обосновать рациональность проектирования данного веб-приложения.

В данной главе производится экономический расчет коммерческой стоимости исследования проекта. Расчеты учитывают расходы на создание и внедрение информационной системы.

Веб-приложение, разрабатываемое и продвигаемое при помощи SEO-продвижения в дипломной работе, предназначен для автоматизации процесса конвертации казахских текстов и документов, содержащих казахские тексты, с кириллицы на латиницу.

Учитывая явление увеличения пользователей интернета, перспективность и не большую величину затрат на использование интернет-ресурсов, вопрос создания веб-приложения становится более актуальным, т.к. имеет ряд преимуществ. Помимо всего этого, следует взять во внимание, что веб-приложение затрагивает тематику государственного масштаба, поэтому имеет довольно широкий охват пользователей. Свое применение веб-конвертер может иметь в государственных, частных, образовательных, административных и любых других учреждениях, которые ведут документацию на государственном языке.

Не требует наличия больших рабочих площадей, и большого количества персонала, не имеет ограничений по зоне действия, т.к. потенциальный пользователь имеет доступ к веб-приложению из любой точки мира, не требует мощной аппаратуры или специфичного программного обеспечения.

Веб-приложение должно быть легким в использовании, иметь понятный функционал, т.к. целевой аудиторией являются не профессиональные IT-специалисты, а простые пользователи интернета.

Для увеличения эффективности и расширения охвата использования веб-приложения необходимо SEO-продвижение. При помощи SEO-продвижения приложение получает лучшие позиции в поисковых системах, что в свою очередь влияет на посещаемость.

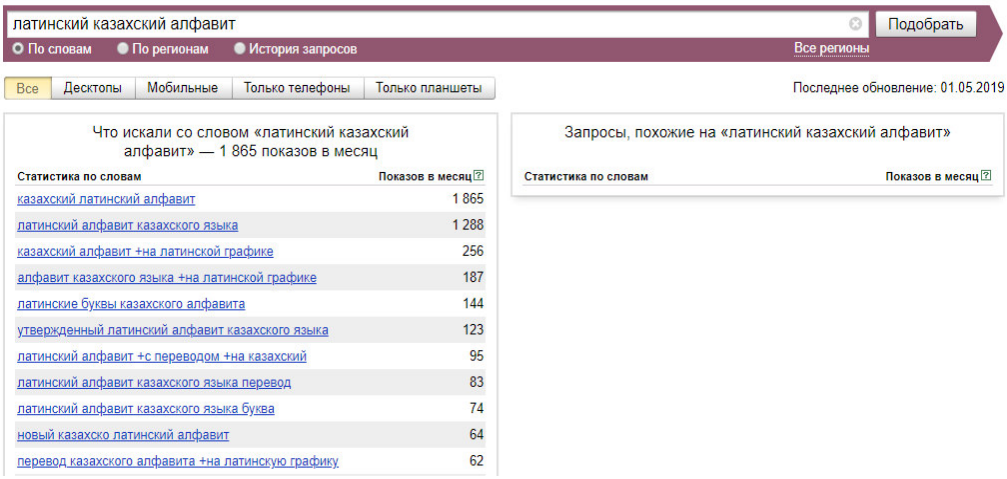


Рисунок 3.1 - Статистика поисковых запросов в wordstat.yandex

Исходя из статистики, за последний месяц в одной из поисковых систем совершали поиск по фразе «латинский казахский алфавит» 1865 человек. В результате успешного SEO-продвижения все пользователи интернета, по данному запросу, будут видеть веб-конвертер.

### 3.1 Расчет трудоемкости разработки программного продукта

Для реализации проекта необходимы финансовые, материальные и трудовые ресурсы;

Для определения трудоемкости разработки приведен перечень всех основных этапов и видов работ, которые должны быть выполнены.

Форма разделения работ по этапам с указанием трудоемкости их выполнения приведена в таблице 4.1.

Таблица 3.1 - Трудоемкость этапов работ

| Этап разработки ПП | Вид работы на данном этапе | Трудоемкость разработки ПП, ч. |
|---|---|---|
| 1. Техническое задание | 1. Постановка задачи<br>2. Сбор материалов и анализ существующего программного обеспечения<br>3. Определение требований к системе | 3<br>3<br>1 |
| 2. Эскизный проект | Функционал, строение и дизайн веб-приложения | 20 |

*Продолжение таблицы 3.1*

| | | |
|---|---|---|
| 3. Технический проект | 1. Выбор инструментальных средств<br>2. Определение требований к аппаратному обеспечению, в том числе и к хостингу | 1<br>1 |
| 4. Рабочий проект | 1. Разметка таблиц структуры БД<br>2. Программирование<br>3. Тестирование и отладка<br>4. Создание модулей синхронизации с иными сторонними сервисами<br>5. Наполнение контентом<br>6. SEO-продвижение | 1<br>220<br>15<br>25<br>6<br>500 |
| ИТОГО трудоемкость выполнения проекта | | 850 |

Поскольку количество часов активной работы по разработке программного продукта равно 850, а в сутки на разработку выделялось 8 часов, следовательно, срок выполнения проекта равен 106 суток. Для дальнейших расчетов время разработки программного продукта округляем до трех месяцев.

### 3.2 Расчет затрат на разработку программного продукта

Общая сумма затрат на материальные ресурсы ($З_М$) определяется по формуле:

$$З_М = \sum_{i=1}^{n} P_i * Ц_i \quad (3.1)$$

где $P_i$ - расход i-го вида материального ресурса, натуральные единицы;

$Ц_i$ - цена за единицу i-го вида материального ресурса, тг;

i - вид материального ресурса;

n - количество видов материальных ресурсов.

Расчет затрат на материальные ресурсы производится по форме, приведенной в таблице 3.2.

Таблица 3.2 - Затраты на материальные ресурсы

| Наименование материального ресурса | Единица измерения | Количество израсходованного материала | Цена за единицу, тг | Сумма, тг |
|---|---|---|---|---|
| 1. Картридж для принтера | шт. | 1 | 10500 | 10 500 |
| 2. Бумага | пачка | 1 | 400 | 400 |
| ИТОГО затраты на материальные ресурсы | | | | 10 900 |

Общая сумма затрат на электроэнергию ($З_Э$) рассчитывается по формуле:

$$З_Э = \sum_{i=1}^{n} М_i \times К_i \times Т_i \times Ц \qquad (3.2)$$

где $М_i$ - паспортная мощность i-го электрооборудования, кВт;

$К_i$ -коэффициент использования мощности i-го электрооборудования (принят $К_i$=0.7);

$Т_i$ - время работы i-го оборудования за весь период разработки ПП ч;

Ц - цена электроэнергии, тг/кВт×ч;

i - вид электрооборудования;

n - количество электрооборудования.

Затраты на электроэнергию приведены в таблице 3.3.

Таблица 3.3 - Затраты на электроэнергию

| Наименование оборудования | Паспортная мощность, кВт | Коэфф-т использ-я мощности | Время работы оборуд-я для разработки ПП, ч | Цена электро энергии, $\frac{т}{кВт \times ч}$ | Сумма, тг |
|---|---|---|---|---|---|
| 1. ПК Samsung | 0,4 | 0,7 | 500 | 16,5 | 2 310 |
| 2. ПК Apple | 0,2 | 0,7 | 20 | 16,5 | 46,2 |
| 3. Ноутбук HP | 0,2 | 0,7 | 6 | 16,5 | 13,86 |
| 4. Ноутбук Lenovo | 0,2 | 0,7 | 230 | 16,5 | 531,3 |
| 5. Модем | 0,2 | 0,7 | 830 | 16,5 | 1917,3 |
| Итого затраты на электроэнергию | | | | | 4818,66 |

Общая сумма затрат на оплату труда ($З_{ТР}$) определяется по формуле:

$$З_{ТР} = \sum ЧС_i \times T_i \quad (3.3)$$

где $ЧС_i$ - часовая ставка i-го работника, тг;

$T_i$ - трудоемкость разработки ПП, чел.×ч; i - категория работника;

n - количество работников, занятых разработкой ПП.

Часовая ставка программиста составляет 800 (тг/ч), трудоемкость разработки – 220 ч.

Часовая ставка SEO-специалиста 500 (тг/ч), трудоемкость – 500 ч.

Часовая ставка WEB-дизайнера 800 (тг/ч), трудоемкость – 20 ч.

Часовая ставка контент-менеджера 400 (тг/ч), трудоемкость – 6.

Часовая ставка научного руководителя составляет 1000 (тг/ч), трудоемкость – 75 ч.

$З_{ТР}$ = 800*220+500*500+800*20+400*6+1000*75 = 519400тг.

Затраты на оплату труда приведены в таблице 4.4.

Таблица 3.4 - Затраты на оплату труда

| Категория работника | Квалиф-ция | Трудоемкость разработки ПП, ч | Часовая ставка, тг/ч | Сумма, тг |
|---|---|---|---|---|
| 1. Программист | Senior-developer | 220 | 800 | 176 000 |
| 2.SEO-специалист | | 500 | 500 | 250000 |
| 3. WEB - дизайнер | | 20 | 800 | 16000 |
| 4. Контент-менеджер | | 6 | 400 | 2400 |
| 5. Научный руководитель | Руководитель проекта | 75 | 1000 | 75000 |
| ИТОГО затраты на оплату труда | | | | 519400 |

Сумма годовых амортизационных отчислений определяется по формуле:

$$А = Перв.стоимость * Норма амортизации/100, \quad (3.4)$$

Амортизационные отчисления приведены в таблице 3.5.

Таблица 3.5 - Амортизация основных фондов (ОФ)

| Наименование оборудования и ПО | Стоимость оборудования и ПО, тг. | Годовая норма амортизации, % | Сумма амортизации в год, тг. | Сумма амортизации за 3 месяца, тг. |
|---|---|---|---|---|
| Модем D-LINK | 4000 | 40 | 1600 | 400 |
| ПК Samsung | 150000 | 40 | 60000 | 15000 |
| ПК Apple | 300000 | 40 | 120000 | 30000 |
| Ноутбук Lenovo | 200000 | 40 | 80000 | 20000 |
| Ноутбук HP | 100000 | 40 | 400000 | 10000 |
| ОС Windows 10 | бесплатно | | | |
| CP Visual Studio | бесплатно | | | |
| ИТОГО амортизация основных фондов | | | | 75400 |

Годовые нормы амортизации ОФ принимаются по налоговому кодексу РК или определяются, исходя из возможного срока полезного использования ОФ:

$$H_{A_i} = \frac{100}{T_{N_i}} \qquad (3.5)$$

Сумма амортизации за один месяц = А / 12.

Сумма амортизационных отчислений за три месяцев равна 75400тг.

В статью «Прочие затраты» включаются расходы на арендную плату, включая коммунальные платежи, затраты на лицензирование и сертификацию, расходы на рекламу, канцелярские и прочие хозяйственные расходы.

Стоимость аренды помещения на месяц равна 44000 тг. (в эту сумму включены коммунальные услуги).

Арендная плата рассчитывается по формуле:

$$АП = Са * S , \qquad (3.6)$$

где Са – срок аренды;

S – стоимость аренды за 1 месяц.

$$АП = 40\ 000 * 3 = 120\ 000 \text{ тг.}$$

Расходы на интернет, месячная оплата которого составляет 4500 тг равны:

$$Р_и= 3 * 4\ 500= 13500 тг.$$

Оплата услуг хостинг компании, предоставляющей аренду дискового пространства (5 000 тг/Gb в год) и доменных имен (1800 тг за 1 доменное имя), в данном проекте предусмотрено 3 Gb дискового пространства и 1 доменное имя:

$$Р_х= 5\ 000*3 + 1\ 800=16\ 800 тг.$$

Прочие хозяйственные расходы составляют 5 000 тг.

Прочие затраты = 120 000 + 13500 + 5 000 + 16800 = 155 300тг.

Социальный налог, согласно Налоговому кодексу РК, составляет 9,5 % от затрат на оплату труда. Пенсионные отчисления не облагаются социальным налогом.

$$СО = (З_{тр} - О_п)*0,095,$$

где $О_п$– отчисления в пенсионный фонд, 10% от ФОТ;

$З_{тр}$ – затраты на оплату труда.

Для программиста:

$$Оп = З_{тр} * 10\% = 176\ 000 * 0,1=17\ 600 тг.$$

$$Ос = (176\ 000- 17\ 600) * 0,095 = 15\ 048тг.$$

Для SEO-специалиста:

$$Оп = З_{тр} * 10\% = 250\ 000* 0,1=25\ 000 тг.$$

$$Ос = (250\ 000- 25\ 000) * 0,095 = 21\ 375тг.$$

Для WEB-дизайнера:

$$Оп = З_{тр} * 10\% = 16000* 0,1=1\ 600тг.$$

$$Ос = (16000- 1\ 600) * 0,095 = 1\ 368тг.$$

Для контент-менеджера:

$$Оп = З_{тр} * 10\% = 2400* 0,1= 240тг.$$

$$Ос = (2400- 240) * 0,095 = 205,2тг.$$

Для дип. Рук.:

$$Оп = З_{тр} * 10\% = 75\ 000* 0,1= 7\ 500тг.$$

$$Ос = (75\ 000- 7\ 500) * 0,095 = 6412,5тг.$$

Итого: 44408,7тг.

На основании полученных данных по отдельным статьям в таблице 4.6 приведена смета затрат на разработку ПП

Таблица 3.6 - Смета затрат на разработку ПП

| Статьи затрат | Сумма, тг |
|---|---|
| 1. Материальные затраты, в том числе: | |
| - материалы | 10 900 |
| - электроэнергия | 4818,66 |
| 2. Затраты на оплату труда. | 519400 |
| 3. Отчисления на социальные нужды. | 44408,7 |
| 4. Амортизация основных фондов. | 75400 |
| 5. Прочие затраты. | 155 300 |
| ИТОГО по смете | 810 227,36 |

### 3.3 Определение возможной (договорной) цены программного обеспечения

Величина возможной (договорной) цены ПП должна устанавливается с учетом эффективности, качества и сроков ее выполнения на уровне, отвечающем экономическим интересам заказчика (потребителя) и исполнителя.

Договорная цена (Цд) для прикладных ПП рассчитывается по формуле:

$$Ц_д = З_{нир} * (1 + (Р/100)), \quad (3.7)$$

где $З_{НИР}$ - затраты на разработку ПП (из таблицы 4.6), тг;

Р - средний уровень рентабельности ПП. % (принято 25%).

$$Ц_Д = 810\,227{,}36 * (1 + 0{,}25) = 1\,012\,784{,}2 \text{ тг.}$$

Цена реализации с учетом НДС рассчитывается по формуле:

$$Цр = Цд + Цд * НДС.$$

НДС, согласно Налоговому кодексу РК, составляет 12 %.

$$Ц_Р = 1012784{,}2 + 1012784{,}2 * 0{,}12 = 1\,134\,318{,}304 \text{ тг.}$$

### 3.4 Расчет срока окупаемости программного продукта

Сотрудник, который совершает процесс конвертации казахских текстов вручную, путем набора соответствующего текста, за 8 часовой рабочий день завершает работу по 10 документам, размер каждого из которых – 30стр. Итого 300 страниц документа в день. Веб-приложение совершает конвертацию документа объемом 30 страниц менее, чем за 1 секунду. Если учесть время, которое сотрудник тратит на выбор, загрузку и выгрузку

документа, то на конвертацию 1 документа он тратит 5 минут в день, соответственно за 8 часовой рабочий день он завершит работу по 96 документам.

В результате использования веб-приложения возможна экономия расходов на выплату зарплаты любой компании более чем в 9 раз, т.к. поддержка веб-приложения не требуется, а своим функционалом веб-конвертер сокращает трудозатраты сотрудника почти в 10 раз.

Зарплата сотрудника, который занимается конвертацией документов вручную, составляет 80 000 тг./мес. После введения в эксплуатацию веб-конвертера казахских текстов, трудозатраты сотрудника сокращаются в 10 раз. Соответственно зарплата теперь составляет 8 000тг./мес.

Расчетный срок окупаемости продукта можно найти по формуле:

$$Т_{ок} = С / Э, \quad (3.8)$$

где С - затраты на разработку и внедрение системы, тенге;
П - прибыль, тенге/месяц.

$$Т_{ок} = 1\ 134\ 318{,}304 / 72000 = 15{,}7 \text{ (месяцев)}.$$

В данном случае срок окупаемости проекта составит 15,7 месяца.

### 3.5 Оценка социально-экономических результатов функционирования программного обеспечения

К социально-экономическим показателям функционирования информационной системы является обработка таких факторов как:

– качество процесса управления;

– длительность и сроки проектирования программного продукта;

– расходы на реализацию приложения и эксплуатацию программного продукта;

– количеству разработчиков.

Эффективность программного продукта определяется его качеством и эффективностью процесса разработки и сопровождения. С позиции специалиста эффективность процесса разработки обуславливается отсутствием необходимости в мощном оборудовании. Все что необходимо специалисту для осуществления процесса разработки – компьютер средней мощности с доступом к интернету. С позиции использования ресурсов пользователи не ограничены ничем, кроме необходимости иметь доступ к интернету.

Разработка веб-конвертера казахских текстов с кириллицы на латиницу занимает 1350 часов (2 месяца). Из них 30 часов занимает процесс постановки задачи и проектирование приложения, 1285 - разработка, 15- тестирование и

отладка, отладка и внедрение. В процессе разработки участвуют 5 человек-программист, SEO-специалист, WEB-дизайнер, контент-менеджер, Project manager (дипломный руководитель). Расходы на реализацию программного продукта составляют 1661688тенге. Договорная цена – 2077110тг. Цена реализации- 2326363тг. Сумма эксплуатации программного продукта составляют 4000 тенге в год, т. к. это сумма размещения сайта на хостинге.

Данные вложения обязательно окупятся за счёт значительного социально-экономического результата. Минимизируются трудовые затраты и время обработки результатов.

Высокая эффективность сопровождения веб-приложения обуславливается беспрепятственной расширения его функционала. Например, нет необходимости приостанавливать работу приложения для внесения функционала латинизации EXCEL документов. Перспективность актуальности веб-конвертера поддерживается наличием соответствующих законодательных актов о переходе казахского алфавита на латинскую графику.

Использование конечного программного продукта не требует какой-то особенной подготовки или квалификации. Любой сотрудник, имеющий опыт использования веб-ресурсов, справится с поставленной задачей.

## 4 Безопасность жизнедеятельности

### 4.1 Анализ помещения

Объект дипломной работы – веб-приложение, используемое компанией, которая осуществляет проектную деятельность, сочетающую в себе консалтинговые работы с последующей автоматизацией предприятий на программной платформе «1С: Предприятие 8» фирмы «1С». Следовательно, для обслуживания столь специфического программного обеспечения, компанией используется большое количество электрооборудования, компьютеров, ИБП (источников бесперебойного питания), принтеров, которые питаются от сети с напряжением 220 В [18].

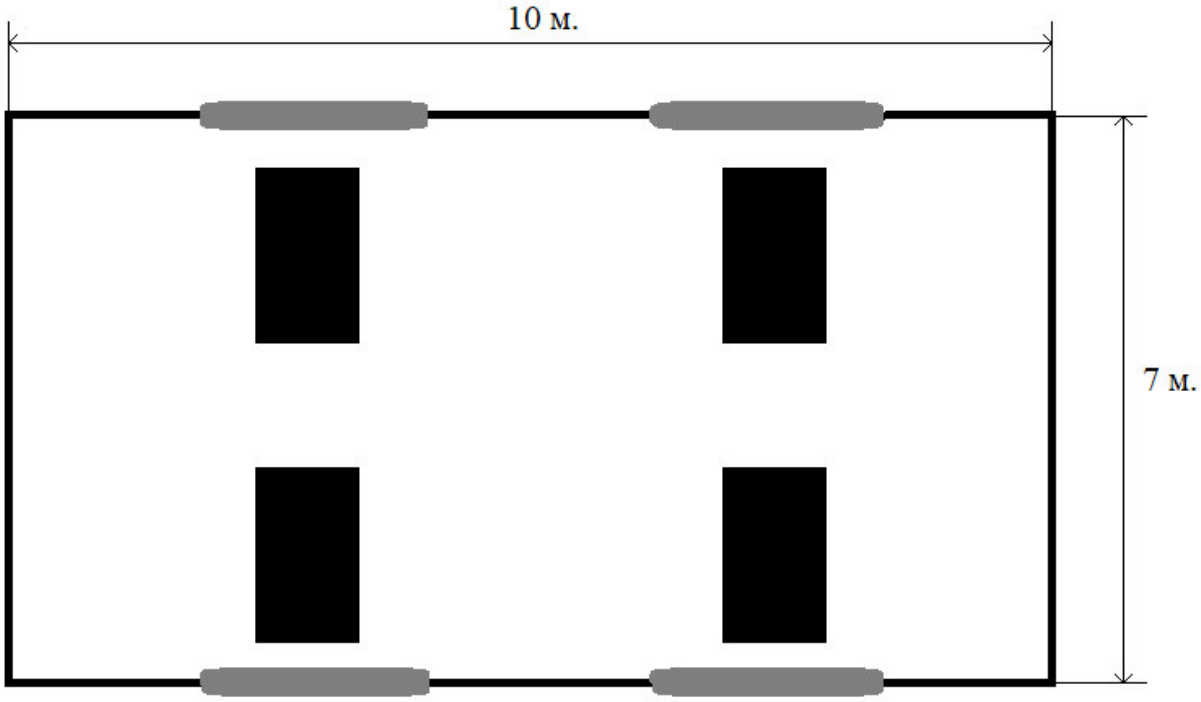


Рис 4.1 – Схема старого помещения

На рисунке 5.1 показана схема помещения.

Площадь помещения равняется 70 м² (ширина – 7м., длина – 10м., высота 4м). В офисе количество сотрудников – 4: IT – специалист (мужчина), консультант (мужчина), бухгалтер (женщина), координатор (мужчина).

Офис находится в городе Алматы. Рассматриваемое помещение находится на 6 этаже. Имеется 4 больших окна, которые не имеют свойства открываться. Окна офиса выходят на ЮВ, поэтому естественного освещения достаточно. Кондиционер отсутствует. Вентилятор старый, советский. Офис не проветривается, так как окна не открываются. Температура в помещении: летом - 26℃, зимой - 18℃.

Необходимо улучшить условия труда, поэтому в технической части раздела БЖД будет рассчитана вентиляция.

### 4.2 Расчет тепловых нагрузок в помещении

В помещениях различного назначения действуют в основном тепловые нагрузки, возникающие снаружи помещения (наружные); а также тепловые нагрузки, возникающие внутри зданий (внутренние) [19].

#### 4.2.1 Теплопоступления и теплопотери в результате разности температур

Данные нагрузки представлены следующими составляющими:

– теплопоступления или теплопотери в результате разности температур снаружи и внутри здания через стены, потолки, полы, окна и двери.

– разность температур снаружи здания и внутри него летом является положительной, в результате чего имеет место приток тепла снаружи во внутрь помещения; и наоборот – зимой эта разность отрицательна и направление потока тепла меняется;

– теплопоступления от солнечного излучения через застекленные площади; данная нагрузка проявляется в форме ощущаемого тепла;

– теплопоступления от инфильтрации.

В зависимости от времени года и времени суток наружные тепловые нагрузки могут быть положительными [20].

Теплопоступления и теплопотери в результате разности температур определяются по формуле (1) [1]:

$$Q_{огр} = V_{пом} \cdot X_o \cdot (t_{Нрасч} - t_{Врасч}), \quad Вт \qquad (4.1)$$

где Vпом – объем помещения, м3;

Vпом=10*7*4=280 м3;

Хо – удельная тепловая характеристика, Вт/м3 0С:

$X_o = 0.42 \quad Вт / м^{3\,0}C$;

tНрасч – наружная температура (параметр А). Для холодного периода – средняя температура самого холодного месяца в 13 часов, для теплого периода – средней температуре самого жаркого месяца в 13 часов;

tВрасч – внутренняя температура, выбирается с учетом комфортных условий или технологических требований, предъявляемых к производственным процессам.

Для теплого времени года:
– tНрасч = 27,6 ℃
– tВрасч = -10 ℃

Qогр = 280*0,42*37.6 = 4422 Вт

Для холодного времени года:
– tНрасч = -20 0С
– tВрасч = 18 0С

Qогр = 280*0,42*(-38) = 4469 Вт

### 4.2.2 Наружные тепловые нагрузки

Избыточная теплота солнечного излучения в зависимости от типа стекла почти до 90% поглощается средой помещения, остальная часть отражается. Максимальная тепловая нагрузка достигается при максимальном уровне излучения, которое имеет прямую и рассеянную составляющие. Интенсивность излучения зависит от широты местности, времени года и времени суток [21].

Расчетные параметры А и Б наружного воздуха даны в таблице 3.

Таблица 4.1 - Расчетные параметры наружного воздуха по СН и П 2.04.05 – 86

| Наименование пункта | Расчет геогр.шир. | Баром. давление гПа | Период года | Параметры А | | | Параметры Б | | | Средне-суточн. амплитуда темпры °С |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Температура °С | Удельн. энтальп. кДж/кг | Скорость ветра м/с | Температура °С | Удельн. энтальп. кДж/кг | Скорость ветра м/с | |
| Алматы | 44 | 930 | Тепл.<br>холод. | 27,6<br>-10 | 51,5<br>-6,7 | 1<br>1,7 | 31,2<br>-25 | 54,4<br>-24,3 | 1<br>1,3 | 11,9<br>- |
| Примечание: Параметр А в теплый период года: среднемесячная температура июля и удельная энтальпия, более высокие значения которых в данном географическом пункте наблюдаются 400 ч и более в году. Параметр Б в холодный период года: ср. температура наиболее холодной пятидневки и соответствующая ей энтальпия. | | | | | | | | | | |

Теплопоступления от солнечного излучения (радиация) определяется по формуле:

$$Q_p = (q^I F_0^I + q^{II} F_0^{II}) \cdot \beta_{с.з.} \qquad (4.2)$$

где $q^I$, $q^{II}$ – тепловые потоки от прямой и рассеянной солнечной радиации, Вт/м2;

$F^I_o$, $F^{II}_o$ – площади светового проема, облучаемые и необлучаемые прямой солнечной радиацией м2;

ßс.з. – коэффициент теплопропускания (согласно талице 4.2 0.15)

Таблица 4.2 - Коэффициенты теплопропускания солнцезащитных устройств ßс.з. (СН и П II, 3-79)

| Солнцезащитные устройства | $ß_{с.з.}$ |
|---|---|
| наружные | |
| - шторы или маркиза из светлой ткани | 0,15 |
| - то же из теплой ткани | 0,20 |
| - ставни-жалюзи, 90° с деревянными пластинками | 0,15 |
| внутренние | |
| - шторы из светлой ткани | 0,4 |
| - то же из темной ткани | 0,8 |

При отсутствии наружных затеняющих козырьков, ребер и т. д. для периода облучения остекления солнцем, когда его лучи проникают через окно в помещение FIo=Fo; FIIo=0:

$$Q_p = q^I F_0 \cdot \beta_{с.з.} = (q_{вп} + q_{вр}) \cdot K_1^c \cdot K_2 \cdot \beta_{с.з.} \cdot n \cdot S_o, \quad Вт, \qquad (4.3)$$

где $q_{вп}$; $q_{вр}$ – тепловые потоки от прямой рассеянной радиации, Вт/м$^2$. По таблице 4.3 для широты в 48$^0$СШ до полудня в 11-12 ч. при расположении Ю:

qвп = 317 Вт/м2; qвр = 88 Вт/м2;

Площадь светового проема:

$$Fo = nSo = 4*3=12 \text{ м2}, \qquad (4.4)$$

где n – число окон;

So – площадь 1 окна.

Таблица 4.3 - Поступление тепла (qвп, qвр) от прямой (П) и рассеянной (Р) радиации в июле через вертикальное остекление (СН и П II-33-75)

| Расчет географ. широта | Истинное солнечное время | | Вертикальное остекление до полудня | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | до полудня | после полудня | С | | ЮВ | | Ю | | ЮЗ | |
| | | | Вертикальное остекление после полудня | | | | | | | |
| | | | С | | ЮЗ | | Ю | | ЮВ | |
| | | | П | Р | П | Р | П | Р | П | Р |
| 44 | 5-6 | 18-19 | 84 | 38 | 72 | 40 | - | 23 | - | 22 |
| | 6-7 | 17-18 | 42 | 70 | 209 | 86 | - | 55 | - | 44 |
| | 7-8 | 16-17 | - | 77 | 333 | 109 | - | 71 | - | 55 |
| | 8-9 | 15-16 | - | 71 | 398 | 108 | 66 | 79 | - | 60 |
| | 9-10 | 14-15 | - | 64 | 387 | 101 | 162 | 81 | - | 63 |
| | 10-11 | 13-14 | - | 60 | 305 | 86 | 245 | 84 | - | 67 |
| | 11-12 | 12-13 | | 59 | 214 | 79 | 288 | 85 | 73 | 77 |
| 48 | 5-6 | 18-19 | 93 | 45 | 95 | 45 | - | 27 | - | 26 |
| | 6-7 | 17-18 | 35 | 69 | 237 | 87 | - | 55 | - | 43 |
| | 7-8 | 16-17 | - | 74 | 363 | 109 | 3 | 73 | - | 53 |
| | 8-9 | 15-16 | - | 70 | 427 | 112 | 80 | 81 | - | 60 |
| | 9-10 | 14-15 | - | 64 | 419 | 107 | 186 | 86 | - | 65 |
| | 10-11 | 13-14 | - | 60 | 352 | 94 | 271 | 87 | 7 | 70 |
| | 11-12 | 12-13 | | 59 | 251 | 84 | 317 | 88 | 106 | 78 |

$К_1$ – коэффициент затемнения остекления переплетами ($К^c_1$ – для облученных проемов; $К_1^Т$ – для проемов в тени) по таблице 4.4:

Таблица 4.4 - Коэффициент К1, учитывающий затемнение световых проемов

| Заполнение светового проема | Незагрязненная атмосфера | Загрязненная атмосфера промрайонов на широте, $^0$С Ш | | | |
|---|---|---|---|---|---|
| | | 44-68 | | 44-68 | |
| | | Проем облучен солнцем $К_1^С$ | | Проем в тени $К_1^Т$ | |
| Стеклоблоки и стеклопрофиль | 1 | 0,7 | 0,75 | 1,6 | 1,75 |
| Остекление в металлических переплетах:<br>- одинарное | 0,8 | 0,56 | 0,6 | 1,28 | 1,40 |
| - двойное | 0,72 | 0,72 | 0,54 | 1,15 | 1,26 |
| Остекление в деревянных переплетах: одинарное | 0,65 | 0,46 | 0,48 | 1,04 | 1,14 |

$К^С_1$= 0.56;

$К_2$ – коэффициент загрязнения остекления по таблице 4.5:

Таблица 4.5 - Коэффициент К2, учитывающий загрязнение остекления для вертикального остекления 80-900.

| Степень загрязнения остекления | $К_2$ |
|---|---|
| Значительное (копоть более 10 мг/м$^3$) | 0,85 |
| Умеренное (копоть 5-10 мг/м$^3$) | 0,9 |
| Незначительное (не более 5 мг/м$^3$) | 0,95 |

$К_2$ = 0.9.

Тогда:

$$Qp = (317+88) * 0,56 * 0,9 * 0,15 * 12 = 367,4 \text{ Вт}$$

Значит общее теплопоступление солнечного излучения с обеих окон равно:

$$Qp = 367,4 + 303,91 = 671,31 \text{ Вт}$$

### 4.2.3 Внутренние тепловые нагрузки

Внутренние нагрузки в жилых, офисных или относящихся к сфере обслуживания помещениях слагаются в основном из тепла:

– выделяемого людьми;

– выделяемого лампами и осветительными, электробытовыми приборами;

– выделяемого компьютерами, печатающими устройствами фотокопировальными машинами пр.

Таблица 4.6 - Тепловыделения человека во внешнюю среду, Вт

| Температура внешней среды $^0$C | Положение сидя | | | Положение стоя либо легкое движение | | | Тяжелая работа | | |
|---|---|---|---|---|---|---|---|---|---|
| | явное | скрытое | общее | явное | скрытое | общее | явное | скрытое | общее |
| 10 | 115 | 15 | 130 | 135 | 21 | 156 | 206 | 84 | 290 |
| 14 | 103 | 15 | 118 | 117 | 21 | 138 | 179 | 84 | 263 |
| 18 | 89 | 15 | 104 | 100 | 33 | 133 | 157 | 93 | 250 |
| 20 | 82 | 21 | 103 | 92 | 42 | 133 | 140 | 110 | 250 |
| 22 | 76 | 26 | 102 | 84 | 48 | 132 | 117 | 132 | 249 |
| 24 | 67 | 35 | 102 | 72 | 60 | 132 | 95 | 154 | 249 |
| 26 | 61 | 41 | 102 | 63 | 69 | 132 | 81 | 168 | 249 |
| 28 | 51 | 51 | 102 | 53 | 79 | 132 | 64 | 185 | 249 |
| 30 | 40 | 60 | 100 | 41 | 89 | 130 | 48 | 198 | 246 |
| 32 | 20 | 78 | 98 | 22 | 106 | 128 | 31 | 213 | 244 |

По таблице 4.6 летом при 26 ℃ один мужчина выделяет явного тепла 61 Вт, а общего – 102 Вт (при работе стоя, легком движении). Женщина выделяет 85% от нормы тепловыделений взрослого мужчины. Тогда выделение явного тепла в помещении составит:

$$Q_л^я=61*3+61*1*0,85=234,85 \text{ Вт.}$$

Значит общее выделение тепла:

$$Q_л^о=102*3+102*1*0,85=392,7 \text{ Вт}$$

По таблице 4.6 зимой при 18 $^0$C один мужчина выделяет явного тепла 89 Вт, а общего – 104 Вт. Женщина выделяет 85% от нормы тепловыделений взрослого мужчины. Тогда выделение явного тепла в помещении составит:

$$Q_л^я=89*3+89*1*0,85=342,65 \text{ Вт.}$$

А выделение общего тепла:

$$Q_л^o=104*3+104*1*0,85=400,4 \text{ Вт.}$$

Теплопоступление от осветительных приборов, оргтехники и оборудования рассчитывается следующим образом. Теплопоступление от ламп определяется по формуле:

$$Q_{осв} = \eta \cdot N_{осв} \cdot F_{пол}, \ \ Вт \ , \qquad (4.5)$$

где η – коэффициент перехода электрической энергии в тепловую (для люминесцентных ламп η=0.5-0.6);
Nосв – установленная мощность ламп (N=40 Вт/м2);
Fпол – площадь пола: Fпол=10*7=70 м2.

Тогда:

$$Qосв=0,55*70*40= 1540 \text{ Вт}$$

Тепло, выделяемое производственным оборудованием, определяется по формуле:

$$Q_{об} = N_{уст} \cdot K \qquad (4.6)$$

$$Qоб=1,5*4*0,7=4,2 \text{ кВт.}$$

### 4.2.4 Расчет теплового баланса помещения

На основании выполненных расчетов составим баланс теплопоступлений в помещении:

Лето: $Q_{изб}$=118+671,31+234,85+1540+4200+1800=8564 Дж

Зима: $Q_{изб}$=4469+671,31+342,65+1540+4200+1800=13023Дж

Так как тепловой баланс для лета больше зимнего теплового баланса, то рассчитаем теплонапряженность воздуха по формуле:

$$Q_н = \frac{Q_{ИЗБ.ЛЕТО} \times 860}{V_{ПОМ}} \ ккал/м3. \qquad (4.7)$$

$$Q_Н=8564*860/280=26,3 \text{ ккал/м}^3$$

При $Q_н$ >20ккал/м$^3$, $\Delta t$ =8 °С,

Определение количества воздуха, необходимое для поступления в помещение:

$$L = \frac{Q_{изб} \times 860}{C \times \Delta t \times \gamma} \quad м3/час \tag{4.8}$$

L=13023*860/(0,24*8*1,206)= 4 836 831,4

где C=0,24 ккал/(кг°С) – теплоемкость воздуха;
γ=1,206 кг/м$^3$ – удельная масса приточного воздуха.

Определение кратности воздухообмена:

$$n = \frac{L}{V_{пом}} \text{ час}^{-1}.$$

n=4 836 831,4/280= 17274,3 час$^{-1}$

### 4.2.5 Выбор кондиционера

Исходя из полученных данных, выберем кондиционер прецизионного типа SUA0501 с верхней подачей в количестве 1 шт.

Таблица 4.7 – Характеристики кондиционера

| Электропитание | Расход воздуха, внутр блок | Расход воздуха, внеш блок | Произв. По теплу | Произв. По холоду | Мощность компрессора | Электронагреватель | Увлажнитель | Расход пара | высота | ширина | глубина | масса |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| В/Ф/Гц | м$^3$/ч | м$^3$/ч | кВт | кВт | кВт | кВт | кВт | Кг/ч | мм | мм | мм | Кг |
| 400/3/5+N | 3020 | 4970 | 17,7 | 16,7 | 4,5 | 6,6 | 2,3 | 3 | 1740 | 1200 | 450 | 260 |

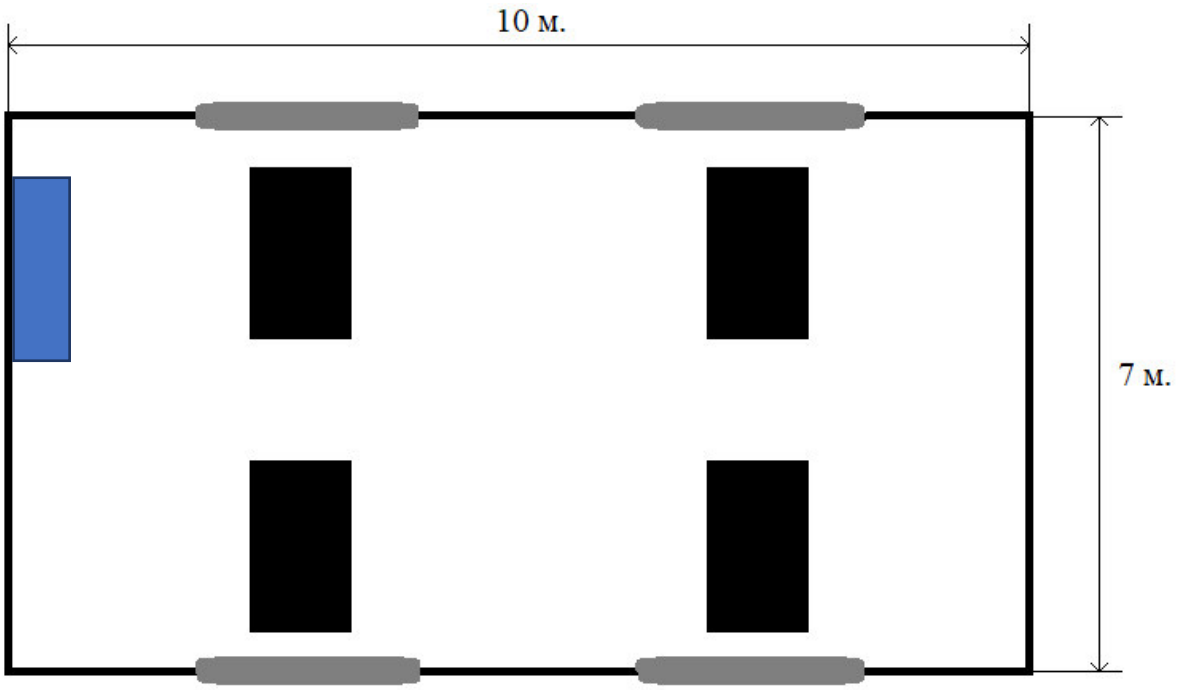


Рисунок 4.2 – Схема расположения кондиционера в офисном помещении

## Заключение

В процессе выполнения работы был изучен рынок современных веб-приложений, посвященных латинизации казахских документов. Проведен сравнительный анализ существующего программного обеспечения и выявлены их недостатки:

– необходимость модернизации алгоритмов, влияющих на скорость работы программного обеспечения;

– необходимость переработки графического пользовательского интерфейса;

– необходимость расширения функциональности приложения в виде создания мультиязычного веб-ресурса.

На основании анализа, а также в соответствии с актуальными вопросами перехода казахского языка на латинскую графику были поставлены задачи, которые в последующем были успешно выполнены, протестированы и введены в эксплуатацию.

Было создано веб-приложение на платформе .NET Core 2.2 в среде разработки Visual Studio и выполнены задачи:

– реализовать функционал латинизации казахских текстов в текстовом поле, а также казахского текста в документах;

– оптимизировать работу приложения с целью ускорения обработки информации;

– реализовать и настроить локализацию и глобализацию веб-приложения для расширения охвата пользователей;

– опубликовать веб-приложение в сети интернет.

Были проведены измерения скорости работы существующего программного обеспечения и созданного в рамках данной дипломной работы веб-конвертера. Для наглядности данные представлены в виде таблицы.

Проведен анализ экономических показателей веб-конвертера, в результате которого выявлено, что внедрение данного веб-приложения сокращает расходы компании на 72000 тг. ежемесячно, а срок окупаемости проекта не превышает 2 лет.

Проведены мероприятия по охране труда в виде расчета аспирационных систем для помещения, в котором используется веб-конвертер.

Оценивая проделанную работу, следует отметить, что веб-приложение является актуальным, так как отвечает современным требованиям рынка программного обеспечения в рамках Республики Казахстан.

Использованные при реализации технологии позволяют беспрепятственно вносить изменения и проводить модернизацию любого структурного элемента веб-конвертера в соответствии с возникающими потребностями. Примером может послужить необходимость расширить функционал до поддержки какого-либо иного формата документов.

## Список литературы


1. Сайт https://docs.microsoft.com/en-us/visualstudio/ide/advanced-feature-overview?view=vs-2019
2. Сайт https://docs.microsoft.com/en-us/visualstudio/get-started/visual-studio-ide?view=vs-2019
3. Сайт https://dotnet.microsoft.com/learn/web/what-is-aspnet-core
4. Сайт https://dotnet.microsoft.com/learn/dotnet/what-is-dotnet
5. Сайт https://dotnet.microsoft.com/languages
6. James Chambers, David Paquette, Simon Timms ASP.NET Core Application Development: Building an application in four sprints (Developer Reference), 2016. – 70с
7. Andrew Lock ASP.NET Core in Action, 2018. – 136с
8. Michael Fitzgerald Introducing Regular Expressions, 2012. – [100-225с]
9. Джозеф А., Бен А. C# 6.0. Карманный справочник, 2015. – 117с
10. Mark J. Price C# 6 and .NET Core 1.0: Modern Cross-Platform Development, 2016. – 216с
11. Сайт https://www.scnsoft.com/blog/web-application-architecture#types-of-web-app-architecture
12. Сайт https://lonewolfonline.net/list-net-culture-country-codes/
13. Сайт https://www.justinmind.com/blog/ui-design-principles-2018-the-new-rules/
14. Сайт https://www.tutorialspoint.com/mvc_framework/mvc_framework_introduction.htm
15. Сайт https://docs.microsoft.com/ru-ru/dotnet/standard/base-types/regular-expression-language-quick-reference
16. Сайт https://www.nuget.org/packages/DocumentFormat.OpenXml/
17. ГОСТ 12.2.032-78 Система стандартов безопасности труда (ССБТ). Рабочее место при выполнении работ сидя. Общие эргономические требования.
18. Сайт https://oventilyacii.ru/ventilyaciya/kratnost-vozduhoobmena-po-snip.html
19. Сайт https://aireng.ru/blog/raschet-estestvennoj-ventilyacii.html
20. Бекишева А.И. Методические указания к выполнению экономической части дипломной работы для бакалавров специальности 5В0703 - Информационные системы – Алматы: АУЭС, 2013
21. Расчет естественной вентиляции: Учебно-методическое пособие «Безопасность жизнедеятельности» выполнение раздела дипломных проектов /Мананбаева С.Е,, Санатова Т.С., Бегимбетова А,С., Бекмуратова Н.К. – Алматы: АУЭС, 2016. – 15с