Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

ОТЧЕТ
по преддипломной практике

Студент Е. О. Кулешов

Руководитель практики
от университета А. М. Ковальчук

Руководитель практики
от предприятия О. А. Самарева

Нормоконтролер А. С. Сидорович

МИНСК 2015

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Данный дипломный проект является системой подачи заявлений на выдачу специальных разрешений на проезд транспортных средств, максимальные весовые и (или) габаритные размеры которых превышают допустимые параметры, установленные для проезда по автомобильным дорогам общего пользования Республики Беларусь.

Данная система предназначена для автоматизации процесса оформления специальных разрешений и предоставления дополнительных возможностей грузоперевозчикам по подаче заявлений на проезд тяжеловесных и крупногабаритных транспортных средств.

Цель разработки – в первую очередь, сокращение сроков оформления специальных разрешений и снижение бумажного документооборота.

Согласно логике работы системы, перевозчику при подаче заявления необходимо предоставить следующую информацию: принадлежность транспортного средства, габаритные параметры, осевые нагрузки, межосевые расстояния, типы осей, вид перевозки, срок перевозки, количество рейсов. Также необходимо задать маршрут, представленный набором точек: начальная, промежуточные и конечная. После чего заявление направляется на рассмотрение в областную группу управления безопасной эксплуатации автомобильных дорог.

Помимо электронной подачи заявления дополнительно система предоставляет следующие возможности:

- автоматизированный процесс расчета платы и оформления специального разрешения по данным электронной заявки, включая проверку соответствия нагрузок и габаритов транспортных средств установленным нормативам, в том числе в период сезонных ограничений;
- ведение электронного реестра поступивших заявлений с текущей информацией о состоянии (рассмотрено, сделан расчет, сделано техническое заключение, не востребовано, оплата расчета);
- проверка поступившей платы по выданным расчетам платы.

По своей структуре система представляет собой веб-приложение, то есть клиент-серверное приложение, в котором клиентом выступает браузер, а сервером – веб-сервер.

Разработка данного приложения производилась по технологии ASP.NET MVC 5, которая в свою очередь является частью платформы Microsoft .NET Framework.

1 ОБЗОР ЛИТЕРАТУРЫ

Первоначально необходимо отметить, что с учетом всевозможных достоинств и недостатков известных технологий, сред разработок, были выбраны следующие компоненты и технологии: для написания веб-приложения используется технология ASP.NET MVC, являющаяся частью платформы Microsoft .NET Framework, язык программирования – C#.

Microsoft .NET Framework – программная технология, предназначенная для создания как обычных программ, так и веб-приложений. Фактически представляет собой операционную систему внутри операционной системы, высокопроизводительную, основанную на стандартах, многоязыковую среду, которая позволяет интегрировать существующие приложения с приложениями и сервисами следующего поколения, а также решать задачи развертывания и использования интернет-приложений. .NET Framework состоит из двух частей: общеязыковой исполняющей среды Common Language Runtime (CLR) и библиотеки классов Framework Class Library (FCL). Основой платформы является виртуальная машина CLR, способная выполнять как обычные настольные программы, так и веб-приложения. Одной из основных идей Microsoft .NET является совместимость различных служб, написанных на разных языках. Например, служба, написанная на C++ для Microsoft .NET, может обратиться к методу класса из библиотеки, написанной на Delphi; на C# можно написать класс, наследованный от класса, написанного на Visual Basic .NET, а исключение, созданное методом, написанным на C#, может быть перехвачено и обработано в Delphi. Каждая библиотека (сборка) в .NET имеет сведения о своей версии, что позволяет устранить возможные конфликты между разными версиями сборок. Так же, как и технология Java, среда разработки .NET создаёт байт-код, предназначенный для исполнения виртуальной машиной. Входной язык этой машины в .NET называется Microsoft Intermediate Language (MSIL), или Common Intermediate Language (CIL), или просто IL. Применение байт-кода позволяет получить кроссплатформенность на уровне скомпилированного проекта (в терминах .NET – сборка), а не только на уровне исходного текста, как, например, в C. Перед запуском сборки в среде исполнения CLR, байт-код преобразуется встроенным в среду JIT-компилятором в машинные коды целевого процессора. Также существует возможность скомпилировать сборку в родной код для выбранной платформы.

Архитектура .NET Framework описана и опубликована в спецификации Common Language Infrastructure (CLI), разработанной Microsoft и

утверждённой ISO и ECMA. В CLI описаны типы данных .NET, формат метаданных о структуре программы, система исполнения байт-кода и многое другое.

Объектные классы .NET, доступные для всех поддерживаемых языков программирования, содержатся в библиотеке FCL. Туда входят классы Windows Forms, ADO.NET, ASP.NET, Language Integrated Query, Windows Presentation Foundation, Windows Communication Foundation и другие. Ядро FCL называется Base Class Library (BCL).

Base Class Library (BCL) – стандартная библиотека классов платформы .NET Framework. Программы, написанные на любом из языков, поддерживающих платформу .NET, могут пользоваться классами и методами BCL – создавать объекты классов, вызывать их методы, наследовать необходимые классы BCL и так далее.

C# – объектно-ориентированный язык программирования. Разработан в 1998-2001 годах в компании Microsoft как один из языков разработки приложений для платформы Microsoft .NET Framework. C# относится к семье языков с C-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java.

Описание базовых принципов функционирования платформы .NET, системы типов .NET и различных инструментальных средств разработки, используемых при создании приложений, базовые возможности языка программирования C#, включая новые синтаксические конструкции, появившиеся с выходом .NET 4.5, а также синтаксис и семантика языка CIL подробно описаны в [1]. В качестве наиболее полного руководства по C#, где описаны базовые конструкции и общие правила языка можно привести [2].

В процессе написания любого программного продукта приходится неоднократно сталкиваться с вопросами использования тех или иных функций. В этом случае незаменима электронная документация к .NET Framework [3], предлагаемая разработчиками компании Microsoft и входящая в пакет Visual Studio. Здесь приведены сведения об использовании классов системных библиотек и поведении конкретных методов.

Наравне с документацией [3], использовалась спецификация языка программирования C# [4], которая по праву считается первоисточником для изучения языка C# и платформы .NET Framework.

ASP.NET – технология создания веб-приложений и веб-сервисов от компании Майкрософт. Она является составной частью платформы Microsoft .NET и развитием более старой технологии Microsoft ASP. На момент своего появления в 2002 году ASP.NET стала огромным шагом вперед.

Стек технологий Microsoft, как он выглядел на то время, показан на рисунке 1.1 [5].

В ASP.NET Web Forms разработчики Microsoft попытались сокрыть как протокол HTTP (с его неизбежным отсутствием состояния), так и язык HTML (который на тот момент был незнаком многим разработчикам), моделируя интерфейс пользователя в виде иерархии серверных объектов, представляющих элементы управления. Каждый такой элемент управления отслеживает собственное состояние между запросами (с помощью средства View State по мере необходимости визуализируя себя в виде HTML – разметки, и автоматически подключая события клиентской стороны (например, щелчки на кнопках) с соответствующим кодом их обработки на стороне сервера). Фактически Web Forms – это гигантский уровень абстракции, разработанный для воссоздания классического, управляемого событиями графического пользовательского интерфейса в веб-среде.

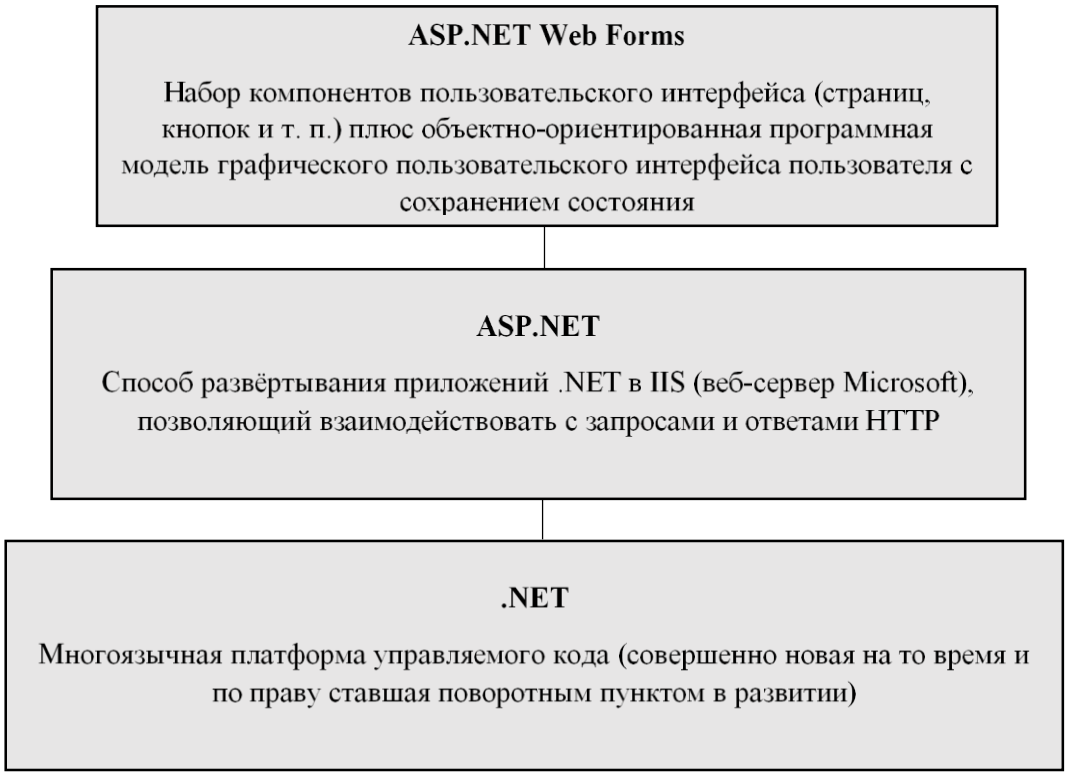


Рисунок 1.1 – Стек технологий ASP.NET Web Forms

Идея состояла в том, чтобы веб-разработка выглядела подобно разработке Windows Forms. Отныне разработчикам не нужно иметь дело с сериями независимых запросов и ответов HTTP; теперь можно мыслить терминами сохраняющего свое состояние интерфейса пользователя. Можно

забыть о веб-среде и её не поддерживающей состояние природе, а вместо этого строить пользовательские интерфейсы с помощью визуального конструктора, использующего технологию перетаскивания, и полагать – или, по меньшей мере, делать вид – что все происходит на сервере.

ASP.NET MVC – это платформа для разработки веб-приложений от Microsoft, которая сочетает в себе эффективность и аккуратность архитектуры «модель-представление-контроллер», новейшие идеи и приемы гибкой разработки, а также все лучшее из существующей платформы ASP.NET. Она представляет собой полномасштабную альтернативу традиционной технологии ASP.NET Web Forms, предоставляя существенные преимущества для всех проектов веб-разработки, кроме наиболее тривиальных. Новая платформа ASP.NET MVC обеспечила радикальный сдвиг в разработке веб-приложений на платформе Microsoft. В ней делается упор на ясную архитектуру, шаблоны проектирования и тестируемость, и не предпринимается попыток сокрытия того, как работает веб-среда.

Термин модель-представление-контроллер (model-view-controller) используется с конца 70-х годов прошлого столетия. Эта модель явилась результатом проекта Smalltalk в компании Xerox, где она была задумана как способ организации некоторых из ранних приложений графического пользовательского интерфейса. Некоторые из нюансов первоначальной модели MVC были связаны с концепциями, специфичными для Smalltalk, такими как экраны и инструменты, но более глобальные понятия все еще применимы к приложениям, и особенно хорошо они подходят для веб-приложений [5].

Концепция паттерна (шаблона) MVC (model-view-controller) предполагает разделение приложения на три компонента [6]:

1. Контроллер (controller) представляет класс, обеспечивающий связь между пользователем и системой, представлением и хранилищем данных. Он получает вводимые пользователем данные и обрабатывает их. И в зависимости от результатов обработки отправляет пользователю определенный вывод, например, в виде представления.

2. Представление (view) – это собственно визуальная часть или пользовательский интерфейс приложения. Как правило, html-страница, которую пользователь видит, зайдя на сайт.

3. Модель (model) представляет класс, описывающий логику используемых данных.

Схема шаблона разработки MVC изображена на рисунке 1.2.

Рисунок 1.2 – Шаблон разработки MVC

В ASP.NET MVC контроллеры являются классами, обычно производными от класса `System.Web.Mvc.Controller`. Каждый метод `public` в классе, унаследованном от класса `Controller`, называется методом действия и посредством системы маршрутизации ASP.NET связан с конфигурируемым URL. Когда запрос отправляется по URL, связанному с методом действия, в классе контроллера выполняются операторы, чтобы провести некоторую операцию по отношению к модели и затем выбрать представление для отображения клиенту. Взаимодействия между контроллером, моделью и представлением показаны на рисунке 1.3 [6].

Рисунок 1.3 – Взаимодействие между контроллером, моделью и представлением

В отличие от ASP.NET Web Forms платформа ASP.NET MVC имеет следующие преимущества [7]:

– она облегчает управление сложными структурами путем разделения приложения на модель, представление и контроллер;

− она не использует состояние просмотра и серверные формы, что делает платформу MVC идеальной для разработчиков, которым необходим полный контроль над поведением приложения;

− она использует схему основного контроллера, при которой запросы веб-приложения обрабатываются через один контроллер, что позволяет создавать приложения, поддерживающие расширенную инфраструктуру маршрутизации;

− она обеспечивает расширенную поддержку разработки на основе тестирования;

− она хорошо подходит для веб-приложений, поддерживаемых крупными коллективами разработчиков, а также веб-разработчикам, которым необходим высокий уровень контроля над поведением приложения.

Платформа ASP.NET MVC предоставляет следующие возможности:

1. Разделение задач приложения (логика ввода, бизнес-логика и логика пользовательского интерфейса), широкое возможности тестирования и разработки на основе тестирования. Все основные контракты платформы MVC основаны на интерфейсе и подлежат тестированию с помощью макетов объекта, которые имитируют поведение реальных объектов приложения. Приложение можно подвергать модульному тестированию без запуска контроллеров в процессе ASP.NET, что ускоряет тестирование и делает его более гибким. Для тестирования возможно использование любой платформы модульного тестирования, совместимой с .NET Framework.

2. Расширяемая и дополняемая платформа. Компоненты платформы ASP.NET MVC можно легко заменить или настроить. Разработчик может подключать собственный механизм представлений, политику маршрутизации URL-адресов, сериализацию параметров методов действий и другие компоненты. Платформа ASP.NET MVC также поддерживает использование моделей контейнера внедрения зависимости (DI) и инверсии элемента управления (IOC). Модель внедрения зависимости позволяет внедрять объекты в класс, а не ожидать создания объекта самим классом. Модель инверсии элемента управления указывает на то, что если один объект требует другой объект, то первые объекты должны получить второй объект из внешнего источника (например, из файла конфигурации). Это облегчает тестирование.

3. Расширенная поддержка маршрутизации ASP.NET. Этот мощный компонент сопоставления URL-адресов позволяет создавать приложения с понятными URL-адресами, которые можно использовать в поиске. URL-адреса не должны содержать расширения имен файлов и предназначены для

поддержки шаблонов именования URL-адресов, обеспечивающих адресацию, оптимизированную для поисковых систем (SEO) и для передачи репрезентативного состояния (REST).

4. Поддержка использования разметки в существующих файлах страниц ASP.NET (ASPX), элементов управления (ASCX) и главных страниц (MASTER) как шаблонов представлений. Вместе с платформой ASP.NET MVC можно использовать существующие функции ASP.NET, например, вложенные главные страницы, декларативные серверные элементы управления, шаблоны, привязку данных, локализацию и другие.

5. Поддержка существующих функций ASP.NET. ASP.NET MVC позволяет использовать такие функции, как проверка подлинности с помощью форм и Windows, проверка подлинности по URL-адресу, членство и роли, кэширование вывода и данных, управление состоянием сеанса и профиля, наблюдение за работоспособностью, система конфигурации и архитектура поставщика.

Платформа ASP.NET MVC предоставляет поддержку для выбора механизмов визуализации. В более ранних версиях MVC использовался стандартный механизм визуализации ASP.NET, который обрабатывал ASPX-страницы с применением оптимизированной версии синтаксиса разметки Web Forms. В версии платформы MVC 3 был введен механизм визуализации Razor, который использует совершенно другой синтаксис. Visual Studio обеспечивает поддержку средства IntelliSense для обоих механизмов визуализации, максимально упрощая внедрение и ответ на данные представления, предоставленные контроллером.

При обработке запросов фреймворк ASP.NET MVC опирается на систему маршрутизации, которая сопоставляет все входящие запросы с определенными в системе маршрутами, которые указывают какой контроллер и метод должен обработать данный запрос. Встроенный маршрут по умолчанию предполагает трехзвенную структуру: контроллер/действие /параметр [7].

Одной из наиболее популярных книг по ASP.NET MVC является [5]. Данная книга даёт подробное описание архитектуры ASP.NET MVC 5, а также инфраструктуры ASP.NET MVC Framework.

В проекте активно используется взаимодействие с базой данных Oracle. Однако использование реляционной базы данных для хранения объектно-ориентированных данных приводит к семантическому разрыву, заставляя писать программное обеспечение, которое должно уметь как обрабатывать данные в объектно-ориентированном виде, так и уметь сохранить эти данные

в реляционной форме. Эта постоянная необходимость в преобразовании между двумя разными формами данных не только сильно снижает производительность, но и создает трудности при программировании, так как обе формы данных накладывают ограничения друг на друга.

Одним из решений проблемы семантического разрыва при использовании реляционной базы данных является Object-Relational Mapping (ORM) – технология программирования, которая связывает базы данных с концепциями объектно-ориентированных языков программирования, создавая «виртуальную объектную базу данных». В качестве ORM-решения в проекте используется NHibernate. Это бесплатная библиотека с открытым исходным кодом, являющаяся портом на .NET популярной на платформе Java библиотеки Hibernate. NHibernate позволяет отображать объекты бизнес-логики на реляционную базу данных. По заданному XML-описанию сущностей и связей NHibernate автоматически создает SQL-запросы для загрузки и сохранения объектов [8]. Помимо официальной документации по Nhibernate [9], хорошим руководством является [10].

СПИСОК ЛИТЕРАТУРЫ

[1] Рихтер, Д. CLR via C#. Программирование на платформе Microsoft .NET Framework 4.5 на языке C# / Д. Рихтер. – СПб. : Питер, 2013. – 896 с.

[2] Шилдт, Г. C# 4.0. Полное руководство / Г. Шилдт. – М. : Вильямс. – 1056 с.

[3] Библиотека MSDN [Электронный ресурс]. – Электронные данные. – Режим доступа: https://msdn.microsoft.com/ru-ru/library/default.aspx.

[4] C# Version 5.0 Specification [Электронный ресурс] : Specification / Microsoft Corporation. – Режим доступа: CSharp Language Specification.docx.

[5] Фримен, А. ASP.NET MVC 5 с примерами на C# 5.0 для профессионалов / А. Фримен. – М. : Вильямс. – 736 с.

[6] Руководство по ASP.NET MVC 5 [Электронный ресурс]. – Электронные данные. – Режим доступа: http://metanit.com/sharp/mvc5/.

[7] Общие сведения о ASP.NET MVC [Электронный ресурс]. – Электронные данные. – Режим доступа: https://msdn.microsoft.com/ru-ru/library/dd381412%28v=vs.108%29.aspx.

[8] Википедия [Электронный ресурс]. – Электронные данные. – Режим доступа: https://ru.wikipedia.org/wiki/NHibernate.

[9] NHibernate Reference Documentation [Электронный ресурс]. – Электронные данные. – Режим доступа: http://nhibernate.info/doc/nh/en/.

[10] Schenker, G. Nhibernate 3 Beginner's Guide / G. Schenker. – Packt Publishing, 2011. – 368 с.

ПРИЛОЖЕНИЕ А

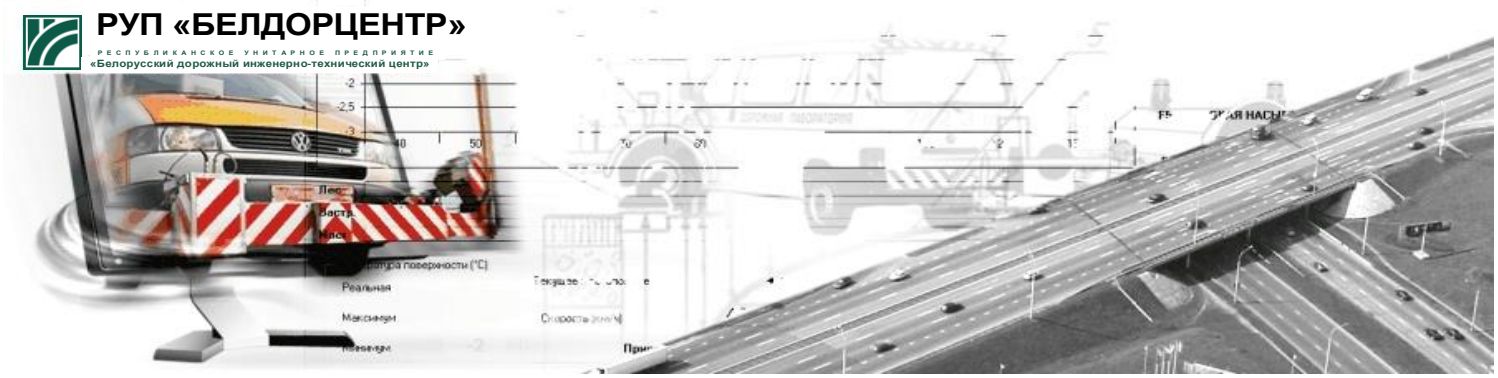
(*справочное*)

Вводный плакат

Автоматизированная система оформления заявлений на выдачу специальных разрешений для проезда тяжеловесных и крупногабаритных транспортных средств

РУП «БЕЛДОРЦЕНТР»

РЕСПУБЛИКАНСКОЕ УНИТАРНОЕ ПРЕДПРИЯТИЕ
«Белорусский дорожный инженерно-технический центр»



Основные функции:

- **Электронная подача заявления на получение специального разрешения на сайте РУП «Белдорцентр»**
- **Ведение электронного реестра поступивших заявлений с текущей информацией о состоянии**
- **Автоматизация процесса оформления специальных разрешений**

Дополнительные возможности:

- **Автоматизированный процесс контроля поступающей информации**
- **Автоматизированный процесс расчета платы**
- **Проверка соответствия нагрузок и габаритов транспортного средства установленным нормативам**

ПРИЛОЖЕНИЕ Б
(*обязательное*)

Структурная схема