МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ им. Р.Е.АЛЕКСЕЕВА

Институт радиоэлектроники и информационных технологий

Разработка веб-приложения для хранения, редактирования и отображения списка фирм по роду их деятельности

Пояснительная записка

к курсовой работе

по дисциплине

\_\_\_\_\_\_\_\_\_\_\_\_\_Распределённая обработка данных \_\_\_\_\_\_\_\_\_\_\_

(наименование дисциплины)

РУКОВОДИТЕЛЬ:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_ Логанов С. В.\_\_

(подпись) (фамилия, и.,о.)

СТУДЕНТ:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_Фараджов Р.Н.О.\_\_

(подпись) (фамилия,и.,о.)

\_\_\_М17-ИСТ-3\_\_\_\_\_\_\_\_

(шифр группы)

Работа защищена «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_

С оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Нижний Новгород 2018

Содержание

[Введение 2](#_Toc535149710)

[1 Выбор и анализ используемых инструментов разработки 3](#_Toc535149711)

[2 Анализ предметной области 4](#_Toc535149712)

[3 Назначение и функциональные возможности 5](#_Toc535149713)

[3.1 Модель прецедентов использования 5](#_Toc535149714)

[3.2 Текстовое описание сценария использования 6](#_Toc535149715)

[4 Структура системы 7](#_Toc535149716)

[4.1 Диаграмма пакетов классов 7](#_Toc535149717)

[4.2 Диаграмма классов моделей 8](#_Toc535149718)

[4.3 Диаграмма классов контроллеров 9](#_Toc535149719)

[4.4 Диаграмма вспомогательного класса 10](#_Toc535149720)

[5 ER-модель предметной области 11](#_Toc535149721)

[6 Диаграмма развертывания веб-приложения 12](#_Toc535149722)

[7 Диаграммы деятельности 13](#_Toc535149723)

[8 Диаграммы последовательности 15](#_Toc535149724)

[Заключение 16](#_Toc535149725)

[Список литературы 17](#_Toc535149726)

[ПРИЛОЖЕНИЕ А 18](#_Toc535149727)

[ПРИЛОЖЕНИЕ Б 19](#_Toc535149728)

[ПРИЛОЖЕНИЕ В 20](#_Toc535149729)

# Введение

Текст

# Выбор и анализ используемых инструментов разработки

Одним из условий по проектированию системы было использование фреймворка ASP.NET MVC. В соответствии с этим требованием проводится дальнейший выбор среды разработки и языка программирования.

Для создания и проектирования веб-приложения в качестве среды разработки было выбрано решение от компании Microsoft – Visual Studio 2017, которая обладает мощными средствами разработки, а также сопровождается хорошей документацией. Эта среда единственная, которая поддерживает разработку под ASP.NET MVC.

Выбор языка С# обусловлен тем, что он является объектно-ориентированным и предназначен для разработки приложений для платформы .NET Framework.

Для разработки клиентской части использовались два языка разметки:

- HTML (Hyper Text Markup Language);

- язык разметки встроенного движка Razor.

Для хранения кода Razor использует файлы формата .cshtml, которые в процессе генерации контроллером ответа с использованием представлений компилируются в классы, из которых затем генерируется страница HTML.

Для добавления стилей к внешнему виду представлений использовался язык CSS (Cascade Style Sheets), который является формальным языком описания внешнего вида документа, написанного с помощью языка разметки.

Также был использован прототипно-ориентированный язык программирования – JavaScript. Он позволил добавить интерактивность во взаимодействия пользователя с веб-страницей.

Выбранный фреймворк реализует паттерн проектирования MVC (Model-View-Controller), который основан на взаимодействии трех компонентов: модель, представление, контроллер. Контроллер принимает на вход запросы с веденными пользователем данными – обрабатывает их, взаимодействуя с моделью и представлением – возвращает пользователю результат обработки запроса. Модель представляет собой логическую структуру организации данных в приложении. Представления служат для интерпретации данных, которые были получены из контроллера, для их отображения в виде элементов пользовательского интерфейса.

Маршрутизация в ASP.NET MVC основана на заданных в системе маршрутах, с которыми происходит сопоставление при обработке запросов пользователя, и в дальнейшем выбирается необходимый контроллер и его метод.

# Анализ предметной области

Исходный веб-ресурс предназначен для отображения списка фирм с разбиением по роду их деятельности. Обычный пользователь может просмотреть список фирм и оставить фидбек. Администратор веб-ресурса может осуществлять просмотр/редактирование/добавление/удаление списка фирм, пользователей системы и фидбеков, которые были оставлены пользователями.

Для отображения взаимосвязей сущностей, определённых в рамках данной работы, была построена диаграмма классов заданной предметной области, приведённая ниже на рисунке 1:

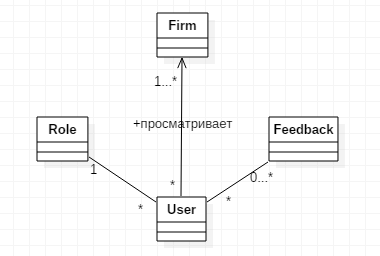


Рисунок 1 – Модель предметной области

Из вышеуказанной диаграммы можно получить следующую информацию:

- каждый User может иметь только одну роль (администратор/рядовой пользователь) [связь многие-к-одному];

- каждый пользователь может оставить множество фидбеков или не оставить вовсе [связь многие-ко-многим];

- каждый пользователь может просмотреть список, содержащий одну или несколько фирм [связь многие-ко-многим].

# Назначение и функциональные возможности

## 3.1 Модель прецедентов использования

Для определения функциональности всей системы используется диаграмма прецедентов. С её помощью можно определить границы моделируемой предметной области, сформулировать модель системы для её последующей детализации.

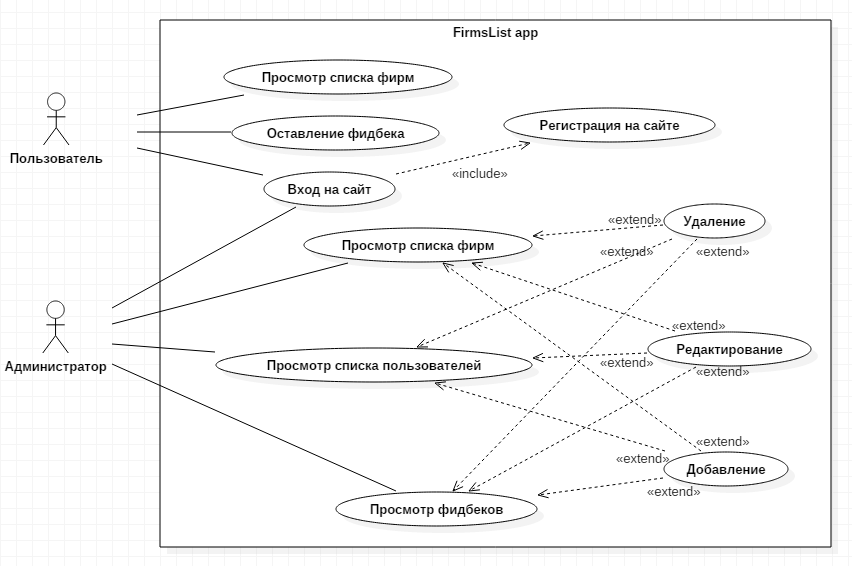
 В исходном проекте диаграмма прецедентов описывает действия, которые могут выполняться пользователями и администратором. Диаграмма прецедентов приведена ниже на рисунке 2:

Рисунок 2 – Диаграмма прецедентов веб-приложения

## 3.2 Текстовое описание сценария использования

Опишем действия администратора со списком пользователей и списком отзывов в данном веб-приложении.

Основной исполнитель – «Администратор»

Цель – Работа со списком пользователей и со списком фидбеков

Основной успешный сценарий:

1. Администратор нажимает на кнопку 'Список пользователей'
2. Администратор переходит на страницу со списком пользователей
3. Администратор нажимает на кнопку 'Просмотреть отзывы'
4. Администратор переходит на страницу со списком фидбеков

Расширенный или альтернативный сценарий:

2.а. Администратор нажимает на кнопку 'Edit’ у какой-либо фирмы

2.а.1. Приложение выдает форму для редактирования информации

2.а.2. Администратор редактирует информацию и сохраняет её, нажав на кнопку ‘Save’

2.а.3. Администратор перенаправляется на страницу со списком фирм

2.б. Администратор нажимает на кнопку 'Delete’ у какой-либо фирмы

2.б.1. Приложение требует от администратора подтверждения операции, нажатием на кнопку ‘Delete’

2.б.2. Администратор подтверждает действие

2.б.3. Приложение удаляет строку с данными о данной фирме

2.б.4. Администратор перенаправляется на страницу со списком фирм

2.в. Администратор нажимает на кнопку 'Details’ у какой-либо фирмы

2.в.1. Приложение выдает информацию о фирме

2.в.2. Пользователь нажимает на кнопку ‘Edit’

2.в.3. Выполняется сценарий под пунктом 2.а.

4.а. Администратор нажимает на кнопку ‘Delete’ у какого-либо отзыва

4.а.1. Приложение требует от администратора подтверждения операции, нажатием на кнопку ‘Delete’

4.а.2. Администратор подтверждает действие

4.а.3. Приложение удаляет строку с данными о фидбеке

4.а.4. Администратор перенаправляется на страницу со списком отзывов

4.б. Администратор нажимает на кнопку 'Details’ у какого-либо отзыва

4.б.1. Приложение выдает информацию об отзыве

# Структура системы

Как было уже указано ранее, выбранный фреймворк реализует паттерн проектирования MVC (Model-View-Controller), который основан на взаимодействии трех компонентов: модель, представление, контроллер, поэтому стоит реализовать диаграммы, которые будут описывать их структуру.

## 4.1 Диаграмма пакетов классов

Для удобства представления всех структуры используется диаграмма пакетов на рисунке 3, где классы веб-приложения распределены в пакеты, отображающие подсистемы:

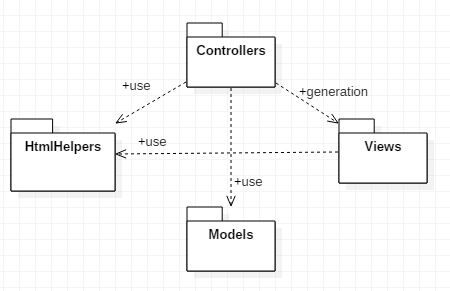


Рисунок 3 – Диаграмма пакетов веб-приложения

На диаграмме выше мы видим, что компонент Controllers взаимодействует со всеми остальными частями отношениями «Use» и «Generation».

В дополнительном компоненте ‘HtmlHelpers’ содержится вспомогательный метод Html, который генерирует Html-разметку для навигационных ссылок.

## 

## 4.2 Диаграмма классов моделей

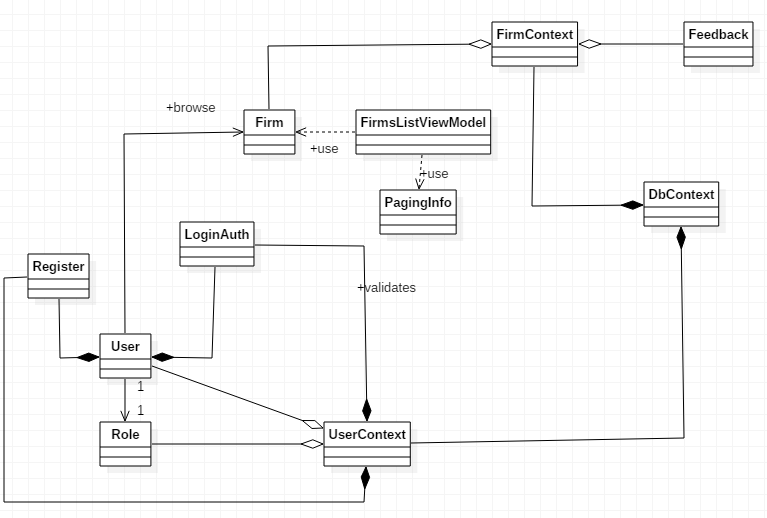
 Для отображения классов, реализующих модели данных, а также взаимосвязей между ними, представлена диаграмма классов пакета моделей на рисунке 4 (построена в UML):

Рисунок 4 – Диаграмма классов пакета моделей веб-приложения

## 

## 4.3 Диаграмма классов контроллеров

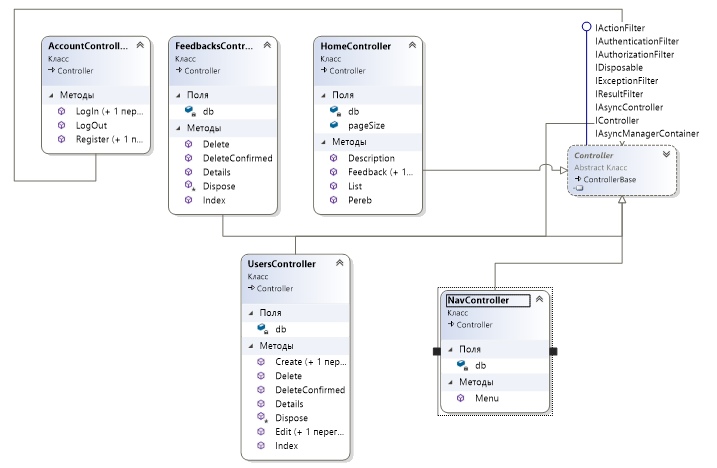
 На рисунке 5 представлена диаграмма классов, разработанная для пакета контроллеров (построена в Visual Studio):

Рисунок 5 – Диаграмма классов пакета контроллеров веб-приложения

## 

## 4.4 Диаграмма вспомогательного класса

Для представления класса вспомогательного пакета ниже представлена диаграмма на рисунке 6:

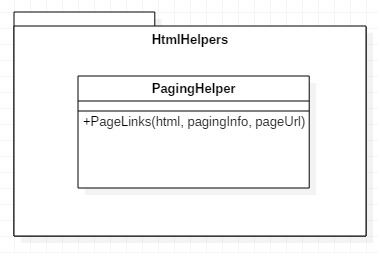


Рисунок 6 – Диаграмма вспомогательного класса веб-приложения

# ER-модель предметной области

Для отображения структуры базы данных веб-приложения на рисунке 7 представлена ER-модель предметной области:

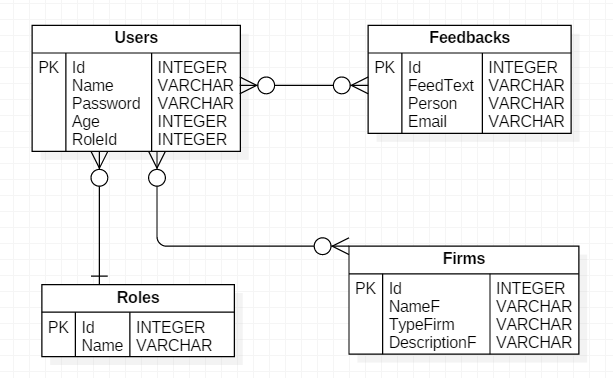


Рисунок 7 – ER-модель предметной области веб-приложения

# Диаграмма развертывания веб-приложения

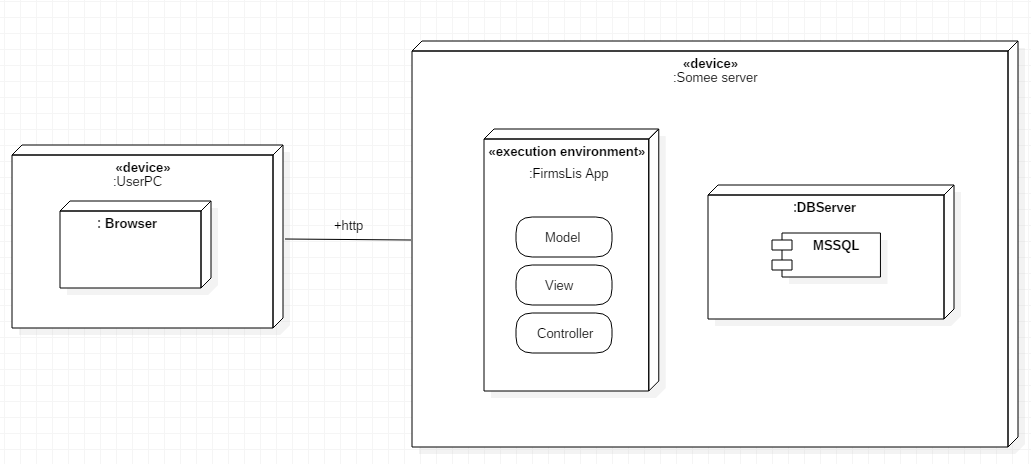
 Для отображения физического представления разработанного веб-приложения на рисунке 8 представлена диаграмма развертывания:

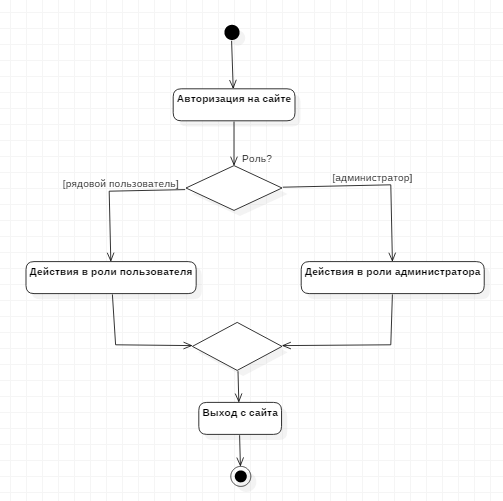
Рисунок 8 – Диаграмма развертывания веб-приложения

# Диаграммы деятельности

При моделировании поведения проектируемой или анализируемой системы возникает необходимость не только представить процесс изменения ее состояний, но и детализировать особенности алгоритмической и логической реализации выполняемых системой операций. Традиционно для этой цели использовались блок-схемы или структурные схемы алгоритмов. Каждая такая схема акцентирует внимание на последовательности выполнения определенных действий или элементарных операций, которые в совокупности приводят к получению желаемого результата.

Диаграммы деятельности считается частным случаем диаграмм состояний. Именно она позволяет реализовать в языке UML особенности процедурного и синхронного управления, обусловленного завершением внутренних деятельностей и действий. Основным направлением использования диаграмм деятельности является визуализация особенностей реализации операций классов, когда необходимо представить алгоритмы их выполнения. При этом каждое состояние может являться выполнением операции некоторого класса либо ее части, позволяя использовать диаграммы деятельности для описания реакций на внутренние события системы.

Поведение системы может быть представлено в виде диаграммы деятельности, приведенной на рисунке 9:

Рисунок 9 – Диаграмма деятельности веб-приложения в целом

На диаграмме показана схема поведения всей системы. Деятельность «Работа в роли администратора» детализирована и выделена в отдельную диаграмму деятельности.

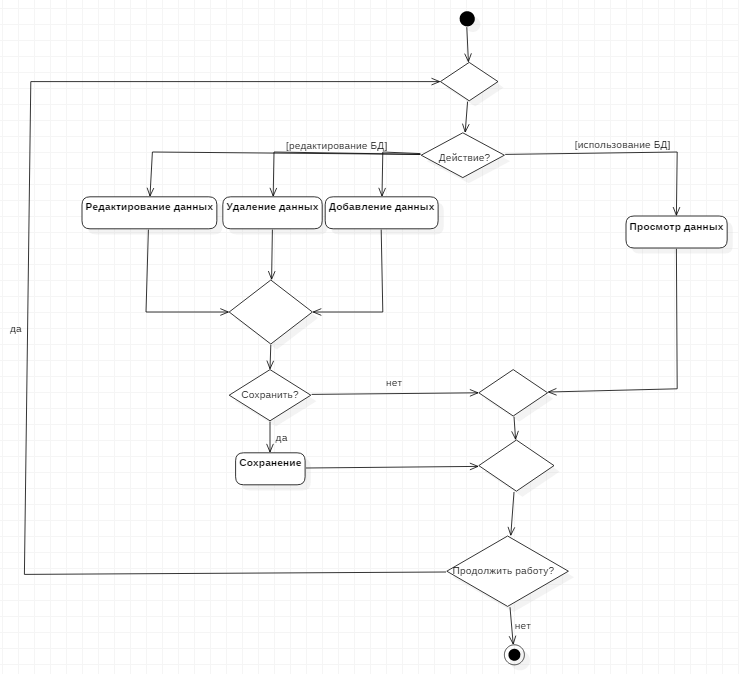
 Поведение системы при работе с ней пользователя, находящегося в роли администратора, представлено на рисунке 10. Там изображены его основные виды деятельности, которые относятся к администрированию и использованию базы данных.

Рисунок 10 – Диаграмма деятельности «Работа в роли администратора»

# Диаграммы последовательности

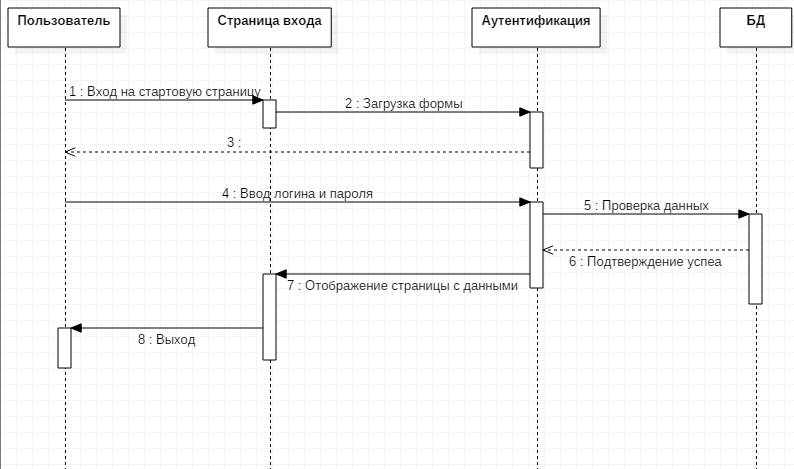
 На диаграмме последовательности «Аутентификация», которая приводится на рисунке 11, изображена последовательность действий пользователя при осуществлении входа в систему.

Рисунок 11 - Диаграмма последовательности «Аутентификация»

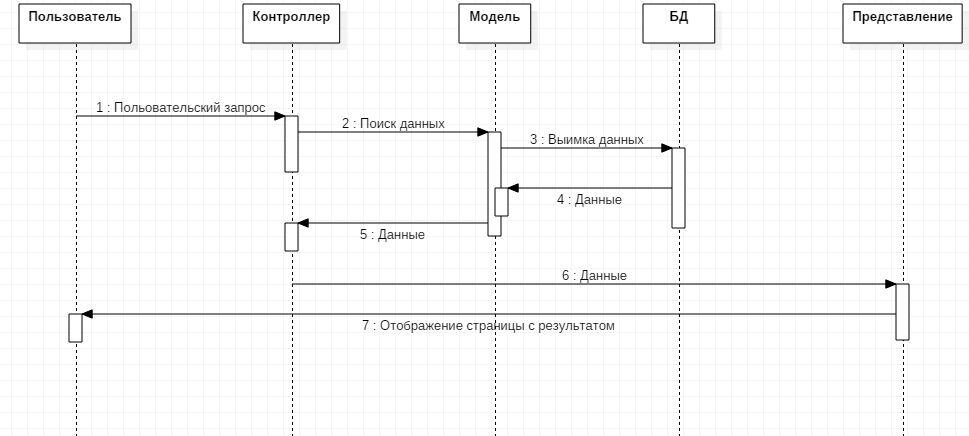
 На рисунке 12 представлена диаграмма последовательности обработки пользовательских запросов, которая показывает, каким образом обрабатываются запросы:

Рисунок 12 – Диаграмма последовательности обработки пользовательских запросов

# Заключение

Текст.

# Список литературы

1. Логанов, С.В. Язык UML и основы объектно-ориентированного проектирования ИС: учеб. пособие / С. В. Логанов; Нижегород. гос. техн. ун-т. - Нижний Новгород, 2017. – 161 с.

# ПРИЛОЖЕНИЕ А

# ПРИЛОЖЕНИЕ Б

# ПРИЛОЖЕНИЕ В