**Министерство образования и науки Российской Федерации**

Федеральное государственное автономное образовательное учреждение

высшего профессионального образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт Кибернетики

Направление подготовки 15.03.06 Информационные системы и технологии

Кафедра вычислительной техники

**Творческий проект**

**Разработка Web-приложения с применением современных Web-технологий**

Выполнил: студент гр. 8И5Б \_\_\_\_\_\_\_ \_\_\_\_\_\_ Смирнов П.О.

Подпись Дата Фамилия И.О.

**Руководитель:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Евсюткин Иван Викторович, программист кафедры ВТ ИК ТПУ |  |  |  |  |  |  |
| ФИО должность,  уч. степень, звание |  | Оценка |  | Дата |  | Подпись |

Комиссия \_\_\_\_\_\_\_ \_\_\_\_\_  Подпись Дата Фамилия И.О.

**Итоговая оценка**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Томск 2017 г.

**Оглавление**

[Введение 3](#_Toc501329032)

[Цель и постановка задачи 3](#_Toc501329033)

[1. ASP.NET MVC 4](#_Toc501329034)

[1.1 ASP.NET Identity 6](#_Toc501329035)

[1.2 Entity Framework 6](#_Toc501329036)

[2. JavaScript 8](#_Toc501329037)

[2.1 JQuery 8](#_Toc501329038)

[2.2 AJAX 8](#_Toc501329039)

[2.3 Bootstrap 9](#_Toc501329040)

[3. REST API 10](#_Toc501329041)

[4. Работа с веб-приложением 11](#_Toc501329042)

[Заключение 15](#_Toc501329043)

[Список литературы 16](#_Toc501329044)

# Введение

Интернет развивается довольно быстро. Быстро растущее число публикаций, посвященных Сети, что предвещает широкое распространение, даже в тех областях, далеких от технологии. Интернет превращается из большой игрушки для интеллектуалов в полноценный источник разнообразной полезной информации для любой категории пользователей.

Через десять лет, эксперты прогнозируют, что около 50 процентов семей (конечно, только в развитых странах) будут использовать его каждый день. Телевидение и радио не будут заменены компьютерами, но будут иметь мощный процессор, большой объем памяти и на самом деле станут в некотором роде компьютерами.

Каждый из нас в настоящее время является пользователем сети и каждый может внести свой вклад в развитие Интернета. Этого достаточно, чтобы создать свой веб-сайт и разместить его в Интернете.

# Цель и постановка задачи

Создать веб-приложение, которое имитирует по своему функционалу денежно-учётную программу. С ее помощью пользователь может контролировать собственные расходы и поступления. Веб-приложение должно реализовывать авторизацию по логину/паролю и предоставлять калькулятор процентов по кредиту для пользователя, а также отобрать курсы валют в режиме онлайн.

Были поставлены следующие задачи:

* изучение методического материала необходимого для создания веб-сайта;
* изучение необходимых языков программирования;
* программная реализация функциональной части веб-сайта;
* проектирование и создание базы данных для работы с файловой системой на сервере и хранения необходимой информации;

# ASP.NET MVC

Для создания веб-приложения была выбрана платформа ASP.NET MVC 5 и необходимая среда разработки для нее Visual Studio 2015. Данная платформа была выбрана исходя из нескольких фактов:

* Основана на .NET и использует язык программирования C#;
* Компилируемый код;
* Используется MVC-паттерн;
* Visual Studio – самое распространённое средство разработки;
* Отличные встроенные инструменты отладки;
* Отображение в формате html-страниц;

Платформа ASP.NET MVC представляет собой фреймворк для создания сайтов и веб-приложений с помощью реализации паттерна MVC.

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

**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.

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

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

Общую схему взаимодействия этих компонентов можно представить следующим образом:



Рисунок 1 – Схема взаимодействий MVC паттерна

В этой схеме модель является независимым компонентом - любые изменения контроллера или представления не затрагивают модель. Контроллер и представление являются относительно независимыми компонентами, и нередко их можно изменять независимо друг от друга.

Благодаря этому реализуется концепция разделение ответственности, в связи с чем легче построить работу над отдельными компонентами. Кроме того, вследствие этого приложение обладает лучшей тестируемостью. И если, допустим, важна визуальная часть или фронтэнд, то можно тестировать представление независимо от контроллера. Либо же можно сосредоточиться на бэкэнде и тестировать контроллер.

Свою реализацию паттерна представляет платформа ASP.NET MVC.

В данном проекте выбрана версия паттерна MVC 5. Хотя во многих аспектах MVC 5 не слишком сильно будет отличаться от MVC 4, многое из одной версии вполне применимо к другой, но в то же время есть и существенные отличия:

* В MVC 5 изменилась концепция аутентификации и авторизации;
* Для создания адаптивного и расширяемого интерфейса в MVC 5 используется css-фреймворк Bootstrap;
* Добавлены фильтры аутентификации, а также появилась функциональность переопределения фильтров;
* В MVC 5 также добавлены атрибуты маршрутизации;
* Использование по умолчанию Entity Framework;

# ASP.NET Identity

ASP.NET Identity – это API-интерфейс, используемый в веб-приложениях для управления данными пользователей, выполнения проверки подлинности и авторизации.

Большинство приложений требуют, чтобы пользователи создавали учетные записи и получали определенные учетные данные для доступа к контенту и услугам. ASP.NET Identity предоставляет средства для выполнения этих операций.

ASP.NET Identity предоставляет инструментарий для работы с пользователями и их аутентификацией. Эта система состоит из библиотеки Microsoft.AspNet.Identity с описанием основных классов и абстракций в виде интерфейсов хранилища пользователей и т.д. Microsoft.AspNet.Identity.EntityFramework — библиотека, которая содержит реализацию пользователя/роли и хранилищ на основании Entity Framework.

Первое, на что хочется обратить внимание, это отсутствие интерфейсов для пользователей и ролей. Identity не диктует нам, каким должен быть наш пользователь. При настройке достаточно указать, какие классы пользователя и роли мы хотим использовать.

# Entity Framework

Entity Framework представляет специальную объектно-ориентированную технологию на базе фреймворка .NET для работы с данными. Если традиционные средства ADO.NET позволяют создавать подключения, команды и прочие объекты для взаимодействия с базами данных, то Entity Framework представляет собой более высокий уровень абстракции, который позволяет абстрагироваться от самой базы данных и работать с данными независимо от типа хранилища.

Центральной концепцией Entity Framework является понятие сущности или entity. Сущность представляет набор данных, ассоциированных с определенным объектом. Поэтому данная технология предполагает работу не с таблицами, а с объектами и их наборами.

Отличительной чертой Entity Framework является использование запросов LINQ для выборки данных из БД. С помощью LINQ можно не только извлекать определенные строки, хранящие объекты, из БД, но и получать объекты, связанные различными ассоциативными связями.

Другим ключевым понятием является Entity Data Model. Эта модель сопоставляет классы сущностей с реальными таблицами в БД.

Entity Framework предполагает три возможных способа взаимодействия с базой данных:

* Database first: Entity Framework создает набор классов, которые отражают модель конкретной базы данных
* Model first: сначала разработчик создает модель базы данных, по которой затем Entity Framework создает реальную базу данных на сервере.
* Code first: разработчик создает класс модели данных, которые будут храниться в БД, а затем Entity Framework по этой модели генерирует базу данных и ее таблицы

В данном проекте выбран подход Code first, т.к. он позволяет наиболее просто определить модель, которая будет хранить описания классов в БД.

# JavaScript

Современные веб-приложения практически невозможно представить без языка клиентской части - JavaScript. Даже при использовании таких серверных языков и технологий, как PHP, ASP.NET, трудно обойтись без JavaScript. Однако чистый JavaScript в реальности используется все меньше. Ему на смену приходят специальные библиотеки, в частности, jQuery. Применительно к ASP.NET MVC при создании веб-приложений библиотеки jQuery играют очень большую роль.

# JQuery

jQuery - это библиотека, которая значительно упрощает и ускоряет написание JavaScript кода.

jQuery позволяет создавать анимацию, обработчики событий, значительно облегчает выбор элементов в DOM и создание AJAX запросов.

Данная библиотека работает со всеми браузерами. Это значит, что не нужно беспокоиться о кроссбраузерной совместимости JavaScript кода.

Для jQuery написано огромное количество плагинов, которые позволяют расширить ее возможности еще больше.

# AJAX

AJAX (Асинхронный JavaScript и XML) представляет собой технологию гибкого взаимодействия между клиентом и сервером. Благодаря ее использованию можно осуществлять асинхронные запросы к серверу без перезагрузки всей страницы.

Применительно к ASP.NET MVC использование AJAX вылилось в целую концепцию под названием "ненавязчивого AJAX" и ненавязчивого JavaScript (unobtrusive Ajax/JavaScript). Смысл этой концепции заключается в том, что весь необходимый код JavaScript используется не на самой веб-странице, а помещается в отдельные файлы с расширением \*.js. А затем в веб-станице ссылается на данный файл кода.

Таким образом отделяется визуализация от логики приложения. Так, выделение скрипта в отдельный загружаемый файл увеличивает производительность сайта, поскольку файл сохраняется в кэше и затем оттуда подгружается.

# Bootstrap

Bootstrap — фреймворк, набор HTML+CSS инструментов и шаблонов для более эффективного и быстрого создания сайтов и веб-приложений.

Фреймворк позволяет быстро произвести макет в HTML и добавить необходимую функциональность

Таким образом, можно выделить основные пункты, почему был выбран данный фреймворк:

* Экономия времени — используются уже готовые классы и дизайн;
* Адаптивность, высокая скорость и оптимизация, стандартизация интерфейсов — динамичные макеты фреймворка качественно отображаются на самых разных устройствах без необходимости внесения изменений в разметку;
* Дизайн — единые шаблоны и стилевое оформление элементов макета и всех страниц на сайте в целом. И при этом Bootstrap кросс-браузерный и хорошо отображается во всех браузерах. Регулярное обновление и дополнение фреймворка самыми современными возможностями HTML и CSS;
* Простота и открытый исходный код;

# REST API

В данном проекте реализуется вывод курса валют, который обновляется в режиме реального времени. Для этого нет необходимости создавать собственную программу, т.к. можно воспользоваться технологией стороннего веб-сервиса.

Веб-сервис — идентифицируемая веб-адресом программная система со стандартизированными интерфейсами.

REST API определяет набор функций, к которым разработчик может совершать запросы и получать ответы. Взаимодействие происходит по протоколу HTTP. Преимуществом такого подхода является широкое распространение протокола HTTP, поэтому REST API можно использовать практически из любого языка программирования.

В отличие от веб-сервисов (веб-служб) на основе SOAP, не существует «официального» стандарта для RESTful веб-API. Дело в том, что REST является архитектурным стилем, в то время как SOAP является протоколом. Несмотря на то, что REST не является стандартом сам по себе, большинство RESTful-реализаций используют стандарты, такие как HTTP, URL, JSON и XML.

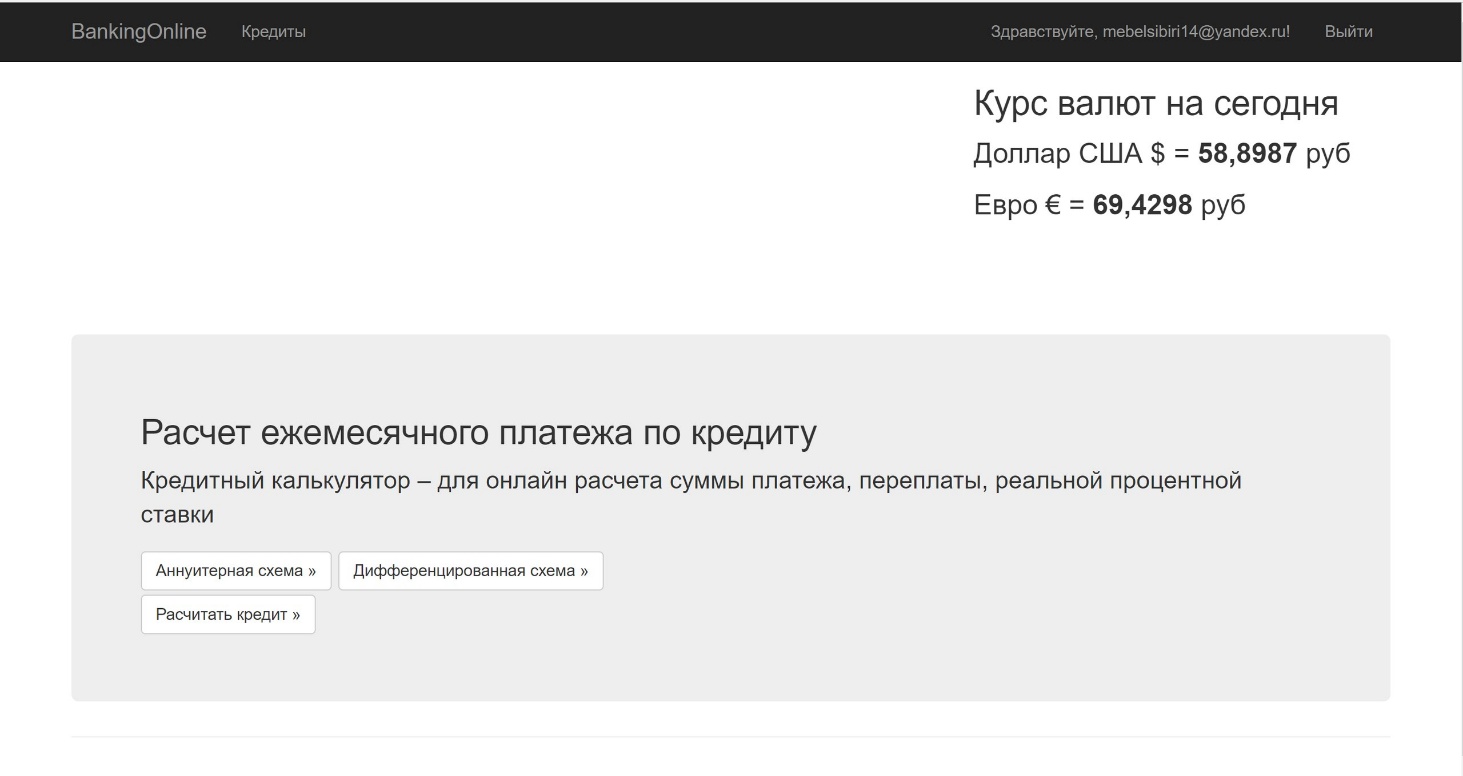
Всё взаимодействие с сервером сводится к 4 операциям:

* получение данных с сервера (обычно в формате JSON, или XML)
* добавление новых данных на сервер
* модификация существующих данных на сервере
* удаление данных на сервере

Операция получения данных не может приводить к изменению состояния сервера.

# Работа с веб-приложением

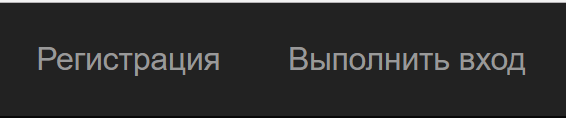
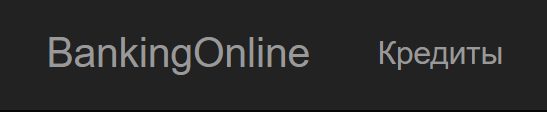
На рисунке 2 можно видеть главную страницу веб-приложения.

Рисунок 2 - Главная страница

Сверху находится панель меню, которая служит для навигации и переходу по страницам веб-приложения, рисунок 3. В нем можно перейти к всем ранее запрошенным кредитам, которые хранятся в БД.

Также на панели меню в правом углу расположена панель авторизации, рисунок 4. Пользователь может создать новый аккаунт, рисунок 5 или же выполнить вход, если регистрировался ранее.

Вся информация о пользователях также храниться в БД.

Рисунок 3 – Панель навигации Рисунок 4 – Панель авторизации

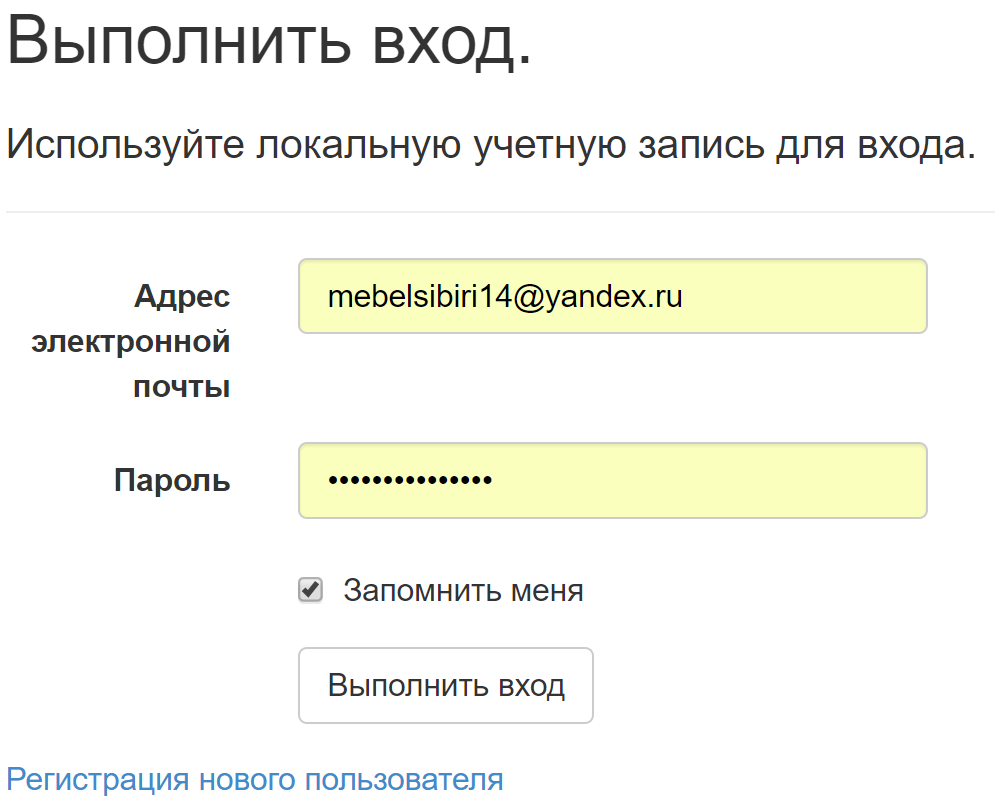


Рисунок 5 – Панель авторизации

Ниже расположено окно веб-сервиса отслеживания курса валют, рисунок 6. Веб-сервис выводит курс валют по отношению к рублю, информация обновляется в режиме реального времени. Веб-приложение асинхронно отправляет AJAX запрос и проверяет курс.

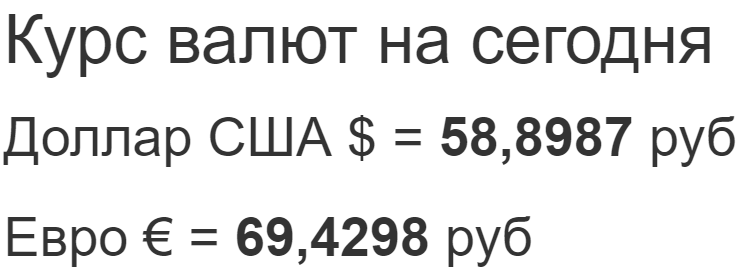


Рисунок 6 – Курс валют

Также на главной странице расположена панель калькулятора валют, рисунок 7. В нем пользователь может произвести расчёт процентов по кредиту двумя способами:

* Аннуитетная схема – равномерное начисление платежей в течение всего срока погашения кредита.
* Дифференцированный платеж — размер ежемесячного платежа по погашению кредита постепенно уменьшается к концу периода кредитования.

Пользователь также может узнать информацию по каждой схеме подробнее, при нажатии на соответствующую кнопку. При нажатии же на кнопку расчёта кредита, выводится форма, где пользователь должен указать информацию по кредиту (проценты, кол-во месяцев, сумма кредита), рисунок 8.

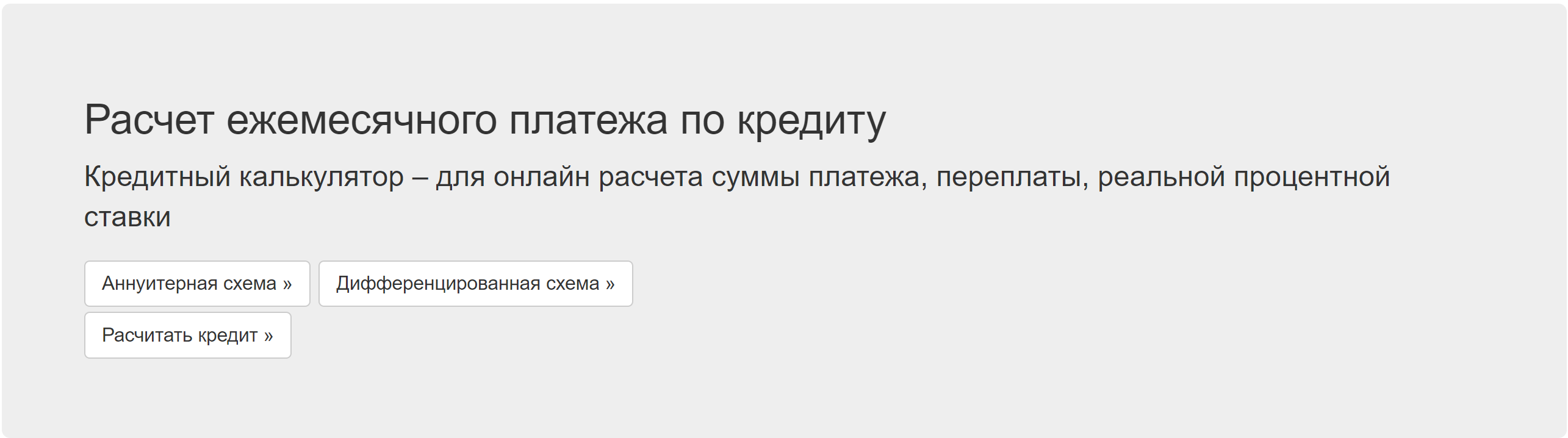


Рисунок 7 – Панель кредитного калькулятора

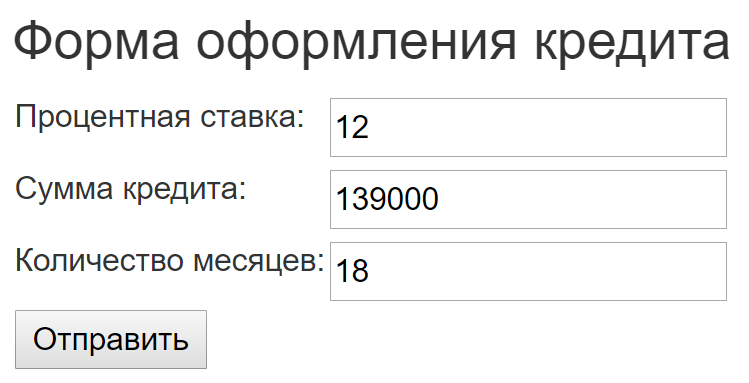


Рисунок 8 – Форма заполнения информации по кредиту

После того как пользователь заполнил форму и отправил данную информацию, происходит переадресация на список всех кредитов пользователя, рисунок 9. По каждому кредиту пользователь может повторно просмотреть рассчитанные проценты и платежи, кликнув на соответствующей вкладке. Если же пользователю больше не требуется информация по какому-то пункту, можно его удалить.

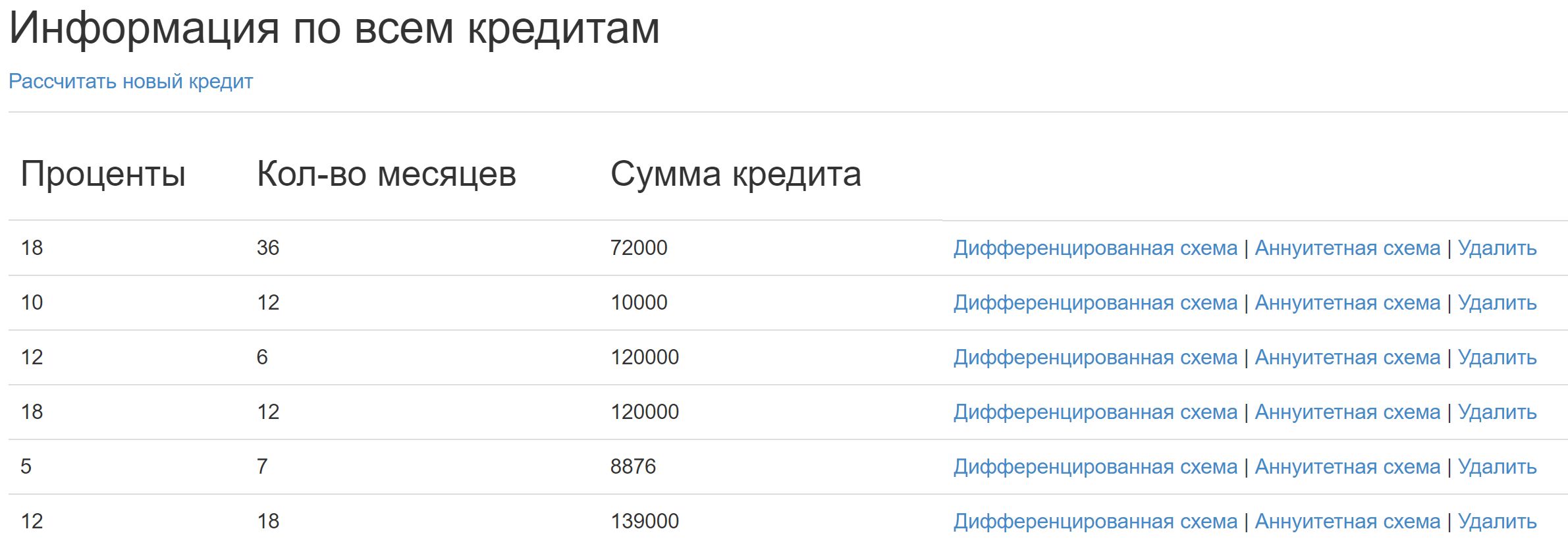


Рисунок 9 – Таблица кредитов

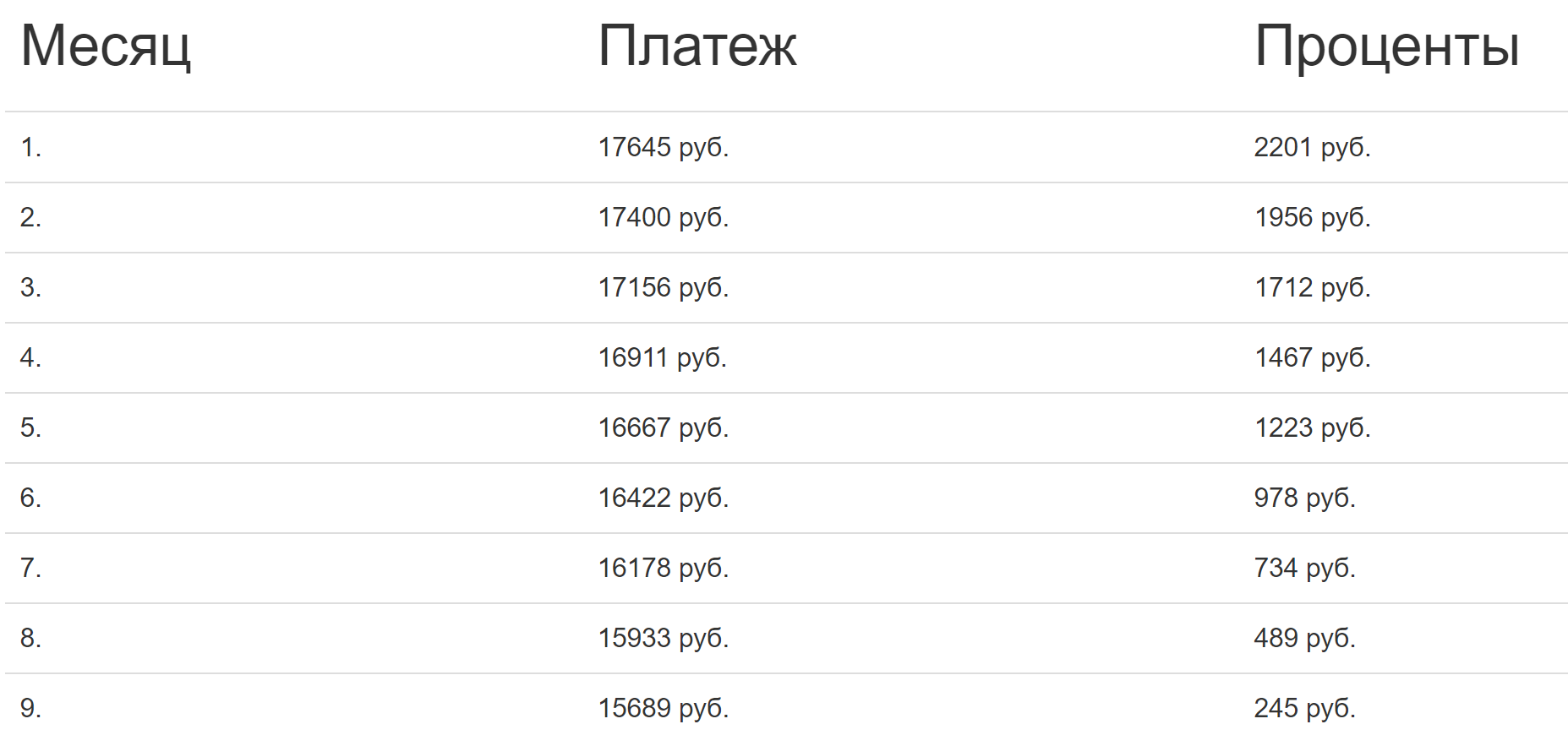


Рисунок 10 – Таблица кредита по дифференцированной схеме

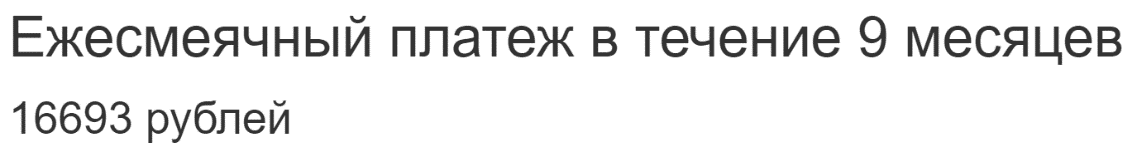


Рисунок 11 – Информация кредита по аннуитетной схеме

# Заключение

В результате выполнения проекта были получены опыт и навыки работы с web-технологиями. Были выполнены все поставленные задачи.

Была изучен методический материал необходимый для создания веб-приложения. В процессе работы над творческим проектом, была изучена платформа ASP.NET MVC. Использовалась библиотека jQuery и технология AJAX, для взаимодействия с клиентской частью. Также был применен css-фреймворк Bootstrap, который позволяет легко и быстро создавать оформление для страниц.

Была произведена работа с сторонними веб-сервисами, которые основаны на таком архитектурном стиле как REST, и их интеграция в приложение.

Созданное веб-приложение является конечным продуктом.

# Список литературы

* <https://ru.wikipedia.org/wiki/>
* <https://metanit.com/sharp/mvc5/1.1.php>
* <https://professorweb.ru/>
* <https://habrahabr.ru/post/38730/>
* <https://learn.javascript.ru/>