**Р Е Ф Е Р А Т**

**ВВЕДЕНИЕ**

Стремление современного человека к материальному благополучию порой приобретает несколько вычурные формы. Желание экономить на покупке дорогих вещей, конечно, вполне оправдано, если только такая экономия не превращается в навязчивую идею и не заставляет человека совершать поступки, о которых, возможно, он впоследствии пожалеет.

Попытки сэкономить там, где другой платит полную цену, активно используются ловкими предпринимателями, исповедующими принцип «цель оправдывает средства». Балансируя на грани дозволенного, подступая к краю законодательства, они предлагают доверчивым обывателям различные способы, соблазняющие возможностью быстро поправить свои финансовые дела.

Последнее время сеть заполонили так называемые скандинавские аукционы, где можно за бесценок приобрести достаточно дорогие вещи. «Скандинавским» этот вид аукционов называется лишь формально - на деле такая система торгов не имеет к Скандинавии никакого отношения. В западных странах этот вид аукционов имеет определённое распространение, но английское название гораздо более точно отражает суть процесса торгов – «Penny bid auction».

Обычный аукцион, как правило, не ограничивает пользователя в размере совершаемой ставки. Скандинавский аукцион похож на обычный лишь терминологией. Основные принципы проведения торгов в этих системах торгов радикально отличаются. Как иногда случается на обычном аукционе, в скандинавском аукционе начальная ставка на тот или иной товар также устанавливается значительно более низкой, чем его рыночная стоимость, но в проведении торгов заложены существенные отличия.

Целью дипломного проекта является создание веб-ориентированного приложения, в котором пользователи смогут разыгрывать товары различного назначения по системе скандинавского аукциона.

**1 АНАЛИЗ ЛИТЕРАТУРЫ**

* 1. **Особенности «Скандинавских» торгов**

Шаг ставки на скандинавском аукционе является фиксированным и составляет очень незначительную величину в сравнении с вероятной стоимостью товара. Скажем, на лот, стоимость которого на рынке составляет 300-400 рублей, стартовая цена может быть установлена всего в 10 рублей, при этом участники торгов имеют возможность повышать ставки всего на 25 копеек.

Следует отметить, что наиболее важное обстоятельство скандинавского аукциона заключается в том, что, делая ставку в 25 копеек, участник должен её оплатить, и плата за ставку составляет 1 рубль. Таким образом, ведя азартную борьбу за лот с другими участниками, сделав, скажем, 20 ставок по 25 копеек, на самом деле участник, даже не выиграв лот, в итоге тратит 20\*1=20 рублей только за возможность делать эти ставки. Прибыль скандинавского аукциона составляет не процент с выигрышной суммы, а именно эта плата за ставку.

Другое важное отличие скандинавского аукциона от обычного состоит в том, что в первом не существует фиксированного периода торгов. По истечении определённого времени каждая сделанная пользователем ставка немного продлевает срок торгов, на период от нескольких секунд до нескольких минут. И, если за это время никто другой не сделает новой ставки, тот, кто сделал последнюю, считается победителем.

**1.2 Сравнение аналогов**

Перед тем, как приступить к реализации проекта, следует проанализировать существующие на данный момент системы подобного характера и тематики. С помощью этого мы можем сделать верные выводы о том, как необходимо создавать собственную систему на основе достоинств и недостатков рассмотренных ресурсов.

Наиболее популярные веб-ориентированные приложения скандинавских аукционов, которые действуют на территории СНГ:

* Gagen.ru – первый российский аукцион, работающий по системе скандинавских аукционов. Есть несколько непродуманных моментов, дисбаланс из-за автоставок и бесплатной раздаче администрацией ставок своим партнерам. Но играть и выигрывать можно. Есть форум, для решения любых вопросов.
* Getbuy.ru – Красивый сайт, много аукционов, но имеется наличие ботов или подставных лиц.
* Vauctione.ru – Молодой и красивый сайт. Недостатки сайта: нет помощи, форума, официального договора публичной оферты.
* Internetlot.ru – Достаточно сырой и не продуманный сайт, яркий пример одной из многих копий Gagen.ru.
* minilot.ru – Копия Gagen.ru, но там организаторы пытаются предоставить честные условия.
* tvoypriz.ru – Неплохой аукцион. Но, полностью отсутствует прозрачность торгов, нет чата или форума. Небольшие технические возможности.
* Goodwin.by – Сайт от белорусских разработчиков. Красивый и понятный дизайн, в сочетании с большими возможностями, тщательной продуманностью торгов, отсутствием ботов и наличии средств коммуникации(форум) делают его лучшим не только в Беларуси, но пожалуй и на всей территории СНГ.

## 1.3 Выбор модели жизненного цикла

Так в проекте требования будут изменяться по ходу разработки, то подходящие модели жизненного цикла приведены ниже.

Таблица 1.1 – Модели ЖЦ

|  |  |
| --- | --- |
| Вид модели | Качество данной модели |
| Инкрементная | + Добавление функции с каждым инкрементом.  - Возможны ситуации, требующие добавления сразу нескольких взаимосвязанных функций. |
| На основе ранее созданных компонентов | + Снижает время разработки.  - Искажаются требования. |
| Спиральная | +Быстрое получение результата.  +Изменяющиеся требования — не проблема.  - Усложнённая структура разработки. |

Из таблицы видно, что наиболее подходящей моделью жизненного цикла является спиральная модель, так как в ходе проекта ожидается возможное изменение требований.

Отличительной особенностью этой модели является специальное внимание к рискам, влияющим на организацию жизненного цикла. Каждый виток спирали соответствует созданию фрагмента или версии программного обеспечения, на нем уточняются цели и характеристики проекта, определяется его качество и планируются работы следующего витка спирали.

**1.4** **Выбор средств разработки**

Для хранения информации наиболее часто используются реляционные системы управления базами данными. Информация в них представлена в виде таблиц. Они содержат простые типы данных и, при необходимости, могут быть взаимосвязаны между собой.

Приложения, как правило, оперируют экземплярами классов (объектами), которые являются абстракциями объектов реального мира. Такой подход более удобен с точки зрения бизнес-логики. Кроме того, это дает такие преимущества при разработке как проверки типов, скорость работы кода и пр.

Таким образом, для использования реляционной базы данных необходимо использовать конвертер, который будет преобразовывать объекты в табличный вид и наоборот. При этом он должен учитывать различия в организации информации в обоих форматах. Например, в таблицах все строки отличаются друг от друга.

Для использования в разработке базы данных была выбрана база данных – MSSQLServer. MSSQLServer имеет прекрасные характеристики по производительности на большом объеме данных, к тому же она имеет большое количество настроек для оптимизации.

В качестве языка программирования баз данных был выбран язык запросов TSQL, т.к. этот язык используется совместно с MSSQLServer.

В качестве языка программирования серверной части был выбран C# и платформа ASP.NET Core которая представляет технологию от компании Microsoft, предназначенную для создания различного рода веб-приложений: от небольших веб-сайтов до крупных веб-порталов и веб-сервисов.

ASP.NET Core является полностью opensource-фреймворком. Все исходные файлы фреймворка доступны на [GitHub](https://github.com/aspnet/).

ASP.NET Core построен на основе кросс-платформенной среды .NET Core, которая может быть развернута на основных популярных операционных системах: Windows, Mac OS X, Linux. То есть веб-приложения могут быть запущены не только на ОС Windows, но и на Linux и Mac OS. А для развертывания веб-приложения можно использовать традиционный IIS, либо кросс-платформенный веб-сервер Kestrel.

Хотя ASP.NET Core преимущественно нацелено на использование .NET Core, но фреймворк также может работать и с полной версией фреймворка .NET.

Благодаря модульности фреймворка все необходимые компоненты веб-приложения могут загружаться как отдельные модули через пакетный менеджер Nuget. Кроме того, в отличие от предыдущих версий платформы нет необходимости использовать библиотеку System.Web.dll.

ASP.NET Core включает в себя фреймворк MVC, который объединяет функциональность MVC, Web API и Web Pages. В предыдущих версии платформы данные технологии реализовались отдельно и поэтому содержали много дублирующей функциональности. Сейчас же они объединены в одну программную модель ASP.NET Core MVC. А Web Forms полностью ушли в прошлое.

Кроме объединения вышеупомянутых технологий в одну модель в MVC был добавлен ряд дополнительных функций.

Одной из таких функций являются тэг-хелперы (tag helper), которые позволяют более органично соединять синтаксис html с кодом С#.

ASP.NET Core характеризуется расширяемостью. Фреймворк построен из набора относительно независимых компонентов. Может использоваться встроенная реализация этих компонентов, либо можно расширить их с помощью механизма наследования, либо вовсе создать и применять собственные компоненты с собственным функционалом.

Также было упрощено управление зависимостями и конфигурирование проекта. Фреймворк теперь имеет свой легковесный контейнер для внедрения зависимостей, и больше нет необходимости применять сторонние контейнеры, такие как Autofac, Ninject.

В качестве инструментария разработки могут использоваться выпуски Visual Studio, начиная с версии Visual Studio 2015. Кроме того, приложения могут создаваться в среде Visual Studio Code, которая является кросс-платформенной и может работать как на Windows, так и на Mac OS X и Linux.

При разработке в Visual Studio 2015/2017 проекты приложений имеют встроенную поддержку с такими популярными инструментами, как Bower, Grunt, Gulp, который позволяют управлять скриптами JavaScript и стилями CSS, автоматизировать и оптимизировать процесс веб-разработки.

Для разработки клиентской части был выбран фреймворк Angular v4.0.

Angular представляет фреймворк от компании Google для создания клиентских приложений. Прежде всего он нацелен на разработку SPA-решений (Single Page Application), то есть одностраничных приложений. В этом плане Angular является наследником другого фреймворка AngularJS. В то же время Angular это не новая версия AngularJS, а принципиальной новый фреймворк.

Одной из ключевых особенностей Angular является то, что он использует в качестве языка программирования TypeScript.

Angular 2 предоставляет такую функциональность, как двустороннее связывание, позволяющее динамически изменять данные в одном месте интерфейса при изменении данных модели в другом, шаблоны, маршрутизация и так далее.

**1.5** **Спецификация функциональных требований**

**2 МОДЕЛИРОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ**

**2.1 Описание функциональности ПС**

Для работы в системе пользователь должен быть зарегистрирован и авторизирован на сайте для этого есть соответствующие действия в системе. Каждый пользователь может просматривать выставленные товары для аукциона и отфильтровать их отображение по заданным параметрам.

Условно можно сделать разделение пользователя на аукционера и аукциониста. Аукционер – участник аукциона. Чтобы сделать ставку аукционист должен заранее приобрести биды. Аукционист создаёт аукцион, а также он может изменять условия его или отменить проведение аукциона.

На рисунке 2.1 показаны варианты использования системы пользователем

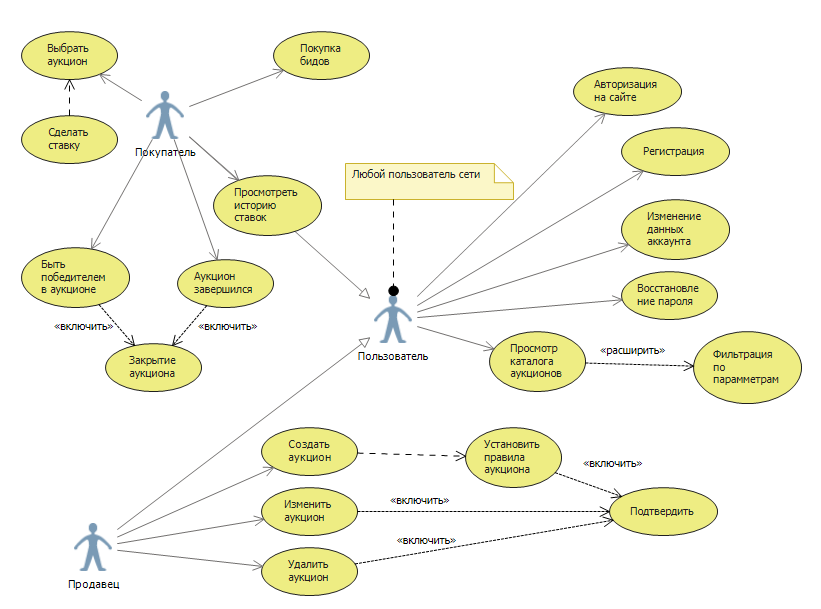


Рисунок 2.1 – UML Use-case -диаграмма взаимодействия пользователя

## 2.2 Спецификация функциональных требований.

Функциональные требования:

* программное средство должно производить автоматический разбор таблиц в базе данных;
* сообщения должны отправляться асинхронно;
* запросы о настройках рассылки должна производиться без перезагрузки страницы в браузере.

Нефункциональные требования:

- вывод сообщения об ошибке;

**3 ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА**

**3.1 Разработка базы данных**

Предметная область содержит следующие сущности:

1. Users - сущность Users является информацией о пользователе, осуществляющем заказ на предоставление услуг автосервиса и прочие действия в процессе пользования услугами сервиса.

Содержит следующие поля:

* Идентификационный номер пользователя
* Логин - email-адрес
* Фамилию пользователя
* Имя пользователя
* Отчество пользователя
* Пароль в зашифрованном виде

1. Roles - сущность Roles является информацией о роли пользователя для разграничения доступа к элементам системы. и содержит идентификационный номер роли и её название.

Содержит следующие поля:

* Идентификационный номер роли
* Название роли

1. Notifications **-** сущность Notifications является информацией об уведомлениях, которые пользователь может получать от сервера. Содержит идентификационный номер и тип уведомления и его содержание.

Содержит следующие поля:

* Идентификационный номер уведомления
* Тип уведомления
* Содержание уведомления

1. Auction – сущность Auction содержит информацию о выставляемом лоте и правила проведения которые задаёт аукционист.

Содержит следующие поля:

* Идентификационный номер аукциона
* Название лота
* Описание лота
* Время создания аукциона
* Время начала аукциона
* Время окончания аукциона
* Путь к картинке выставляемого лота
* Начальная цена лота
* Цена приращения к цене лота за ставку
* Количество бид за ставку
* Цена лота на рынке
* Идентификационный номер аукциониста выставляемого лота
* Идентификационный номер победителя в аукционе
* Цена лота после завершения аукциона

1. AuctionHistory – сущность AuctionHistory содержит информацию о проведённых аукционах.

Содержит следующие поля:

* Идентификационный номер аукциона
* Идентификационный номер победителя в аукционе
* Цена лота после завершения аукциона
* Информация о сделанный ставках в формате JSON следующие поля: идентификационный номер участника аукциона сделавшего ставку, время ставки.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**