**1 ОБЗОР ЛИТЕРАТУРЫ**

**1.1 Обзор аналогов**

В данном подразделе рассмотрим существующие аналоги реализуемого программного обеспечения.

**1.1.1 Thomann.de**

Thomann.de [1] (см. рисунок 1.1) – немецкий специализированный сайт по продаже музыкального оборудования. Главная страница представлена удобным интерфейсом с приятным оформлением, без лишнего нагромождения.

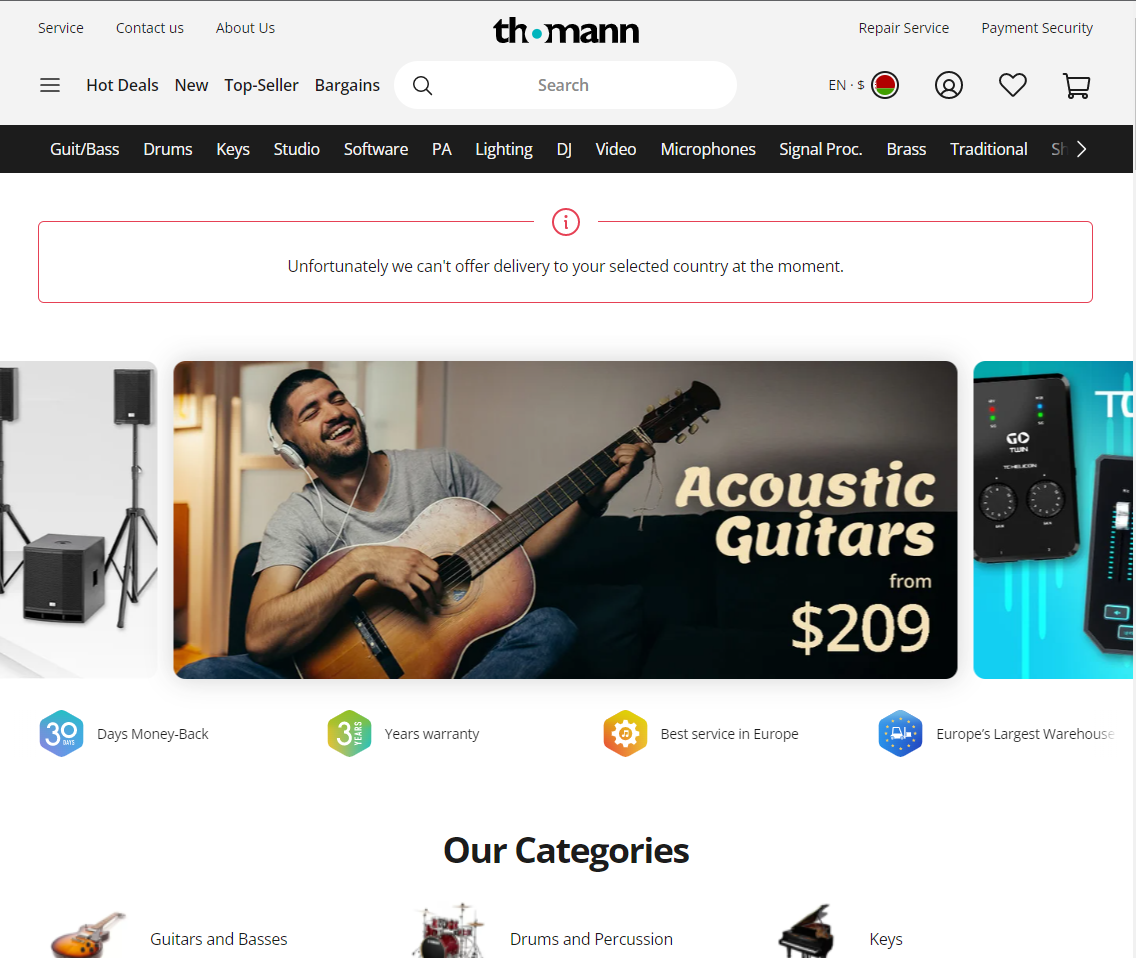


Рисунок 1.1 – Главная страница Thomann.de [1]

На сайте отсутствует явная реклама. Однако на сайте присутствует неуместный раздел – раздел новостей. В случае если пользователь захочет узнать новости по тематике определенного звукового оборудования, то он воспользуется специализированным сайтом. В данном случае функционал раздела новостей является избыточным. На данном сайте можно найти большое количество разнообразной музыкальной техники: от бас-гитар до световых установок. Существенным недостатком является отсутствие локализации для русского языка, что может принести некотором пользователям дискомфорт при пользовании сайтом и оформлении заказов.

Резюмируя вышеперечисленное, можно выделить явные достоинства и недостатки.

Достоинства:

- широкий ассортимент товара;

- практичный интерфейс;

- удобная адаптация для мобильных устройств.

Недостатки:

- отсутствие русской локализации;

- невозможность в данный момент доставки в Республику Беларусь.

**1.1.2 Guitarland.by**

Guitarland.by [2] (см. рисунок 1.2) – сайт магазина музыкальных инструментов. Деятельность данного сайта направлена на более узкий спектр музыкального оборудования.

Здесь в ассортименте представлены инструменты для живого исполнения и записи. Однако кроме нужных разделов на сайте присутствует достаточное количество лишних разделов. Таковыми являются новости и события, которые можно прочитать на новостных сайтах. Вторым таким разделом является статьи Guitarland, который было бы целесообразнее вести в социальных сетях магазина, а на самом сайте размещать ссылки на страницы в социальных сетях. Существенным недостатком данного сайта является отсутствие прямой связи с продавцом напрямую через сайт. Также сайт не оптимизирован для использования в мобильной версии, что ухудшает удобство просмотра сайта.

Исходя из предложенных ранее фактов, резюмируем достоинства и недостатки данного ресурса.

Достоинства:

- узконаправленность (удобство для клиента, который знает чего хочет);

- наличие сезонных скидок на большое количество товаров;

- торговое помещение в центре Минска.

Недостатки:

- отсутствие адаптации для мобильных устройств;

- присутствие большого количества лишней информации на странице сайта.



Рисунок 1.2 – Главная страница Guitarland.by [2]

**1.2 Обзор технологий**

Разрабатываемая в рамках данного дипломного проекта программная система является веб-приложением. Под веб-приложением понимается клиент-серверное приложение, в котором роль клиента играет браузер, а сервером является веб-сервер, который может хранится в облачном хранилище. Логика веб-приложения распределена между сервером и клиентом, хранение данных осуществляется, преимущественно, на сервере, обмен информацией происходит по сети. Благодаря такому подходу клиент может не зависеть от конкретной операционной системы. Из этого следует, что веб-приложения являются кроссплатформенными сервисами.

Разрабатываемая программная система имеет монолитную архитектуру. Концепция монолитного программного обеспечения подразумевает, что различные компоненты приложения объединяются в одну программу на одной платформе. Обычно монолитное приложение состоит из базы данных, клиентского пользовательского интерфейса и серверного приложения. Все функции такого приложения управляются в одном месте.

Для разработки серверной части приложения использовался паттерн MVC. MVC представляет из себя архитектурный шаблон, который означает, что приложение MVC будет разделено на три части, а именно:

- модель;

- представление;

- контроллер.

Модель представляет из себя данные, передаваемые между представлениями и контроллерами, а также модель может содержать операции и преобразования для манипулирования этими данными.

Представления применяются для визуализации части модели в виде пользовательского интерфейса.

Контроллеры обрабатывают поступающие запросы, выполняют операции с моделью и выбирают представления для визуализации пользователю.

В большинстве случаев для разработки веб-сервера используются следующие языки программирования: Java, JavaScript, Python, PHP, C#, Ruby.

**1.2.1 Ruby**

Для написания веб-сервера данного дипломного проекта используется язык программирования Ruby [3]. Ruby – это интерпретируемый, высокоуровневый, динамичный, универсальный язык программирования с открытым исходным кодом, который фокусируется на простоте и производительности. Он был спроектирован и разработан в середине 1990-х годов Юкихиро Мацумото в Японии.

Ruby – это язык динамического программирования со сложной, но выразительной грамматикой и базовой библиотекой классов с богатым и мощным API. Ruby - вобрал в себя черты таких языков, как Lisp, Smalltalk и Perl, но использует грамматику, которой без особого труда смогут овладеть программисты, работающие на языках С и Java.

Ruby является абсолютным объектно-ориентированным языком, но в нем также неплохо уживаются процедурные и функциональные стили программирования. Приоритетом Ruby является удобство и минимизация затрат труда программиста при разработке программы, освобождение программиста от рутинной работы, которую компьютер может выполнять быстрее и качественнее. Особое внимание, в частности, уделено будничным рутинным занятиям (обработка текстов, администрирование), и для них язык настроен особенно хорошо. В противоположность машинно-ориентированным языкам, работающим быстрее, Ruby – язык, наиболее понятный человеку, даже очень далекому от программирования. Любая работа с компьютером выполняется людьми и для людей, и необходимо заботиться в первую очередь о затрачиваемых усилиях людей.

**1.2.2 JavaScript**

Для разработки пользовательского интерфейса используется язык программирования JavaScript и библиотека jQuery.

JavaScript работает на клиентской стороне приложения и используется для определения того, как веб-страницы ведут себя при возникновении события [!]. JavaScript является простым в изучении, широко используемым, а также мощным языком для управления поведением веб-страниц.

JQuery – легковесная библиотека JavaScript, слоганом которой является “пиши меньше, делай больше”. Цель jQuery - сделать использование JavaScript на проекте намного проще. JQuery берет много общих задач, для выполнения которых требуется много строк JavaScript-кода, и обертывает их в методы, которые можно вызвать одной строкой. JQuery также упрощает многие сложные вещи из JavaScript, такие как вызовы AJAX и манипуляции с DOM элементами [!].

**1.2.3 Ruby on Rails**

Ruby on Rails – фреймворк, написанный на языке программирования Ruby с открытым исходным кодом, реализует архитектурный шаблон Model-View-Controller для веб-приложений. Ruby on Rails включает в себя инструменты, которые облегчают общие задачи разработки «из коробки», такие как scaffolding, которые могут автоматически генерировать некоторые модели и представления, необходимые для базового сайта. В конфигурации по умолчанию модель отображает одну из таблицы в базе данных. Например, класс модели User обычно определяется в файле «user.rb» в каталоге app/models и привязывается к таблице «users» в базе данных.

Контроллер – это компонент Rails, который отвечает на внешние запросы от веб-сервера к приложению, определяя, какой файл отображения нужно отрисовать. Контроллеру также может потребоваться запросить одну или несколько моделей для получения информации и передать их в отображение.

Ruby on Rails также заслуживает внимания за широкое использование библиотек JavaScript для написания Ajax-запросов. Для этого прекрасно подойдёт jQuery, который полностью поддерживается как замена Prototype и поддерживается по умолчанию в Rails с версии 3.1.

Основным преимуществом языка программирования Ruby и фреймворка Ruby on Rails является скорость разработки. На практике скорость разработки проектов на Ruby on Rails выше на 30-40 процентов по отношению к любому другому языку программирования или фреймворку. Такой прирост к скорости разработки объясняется обширным набором готовых к работе штатных инструментов Ruby on Rails, возможностью использовать готовые решения других разработчиков и удобством программирования на Ruby.

Кроме того, в отличие от других фреймворков, в составе Ruby on Rails есть отличные средства автоматизированного тестирования, что ускоряет переход проекта от стадии «программа написана» к стадии «программа работает без ошибок». Зачастую именно этот переход отнимает больше всего времени при реализации практически любого проекта.

**1.2.4 Шаблон проектирования MVC**

Шаблон проектирования MVC [4] (см. рисунок 1.3) – это не столько шаблон, сколько метод или идея организации приложения, его структуры. Сама идея заключается в том, чтобы отделить бизнес логику приложения от его представления. Как следует из названия MVC (Model – View – Controller) состоит из трех основных компонентов: модель, представление и контроллер.

При этом модификация каждого из трех модулей происходит независимо. Данный подход описывает один из принципов SOLID – принцип единой ответственности. Это значит, что каждый объект (модуль) имеет одну ответственность и все поведение объекта должно быть направлено на выполнение только этой одной ответственности.

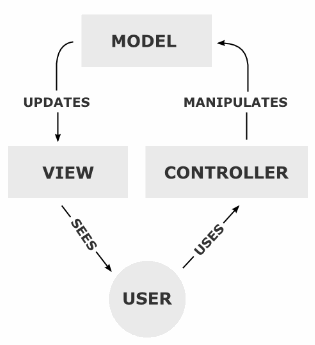
****

Рисунок 1.3 – Шаблон MVC [4]

Модель в шаблоне MVC – основная часть логики, которая отвечает за выборку данных из базы, изменение данных, расчеты. Вообще модель можно представить, как набор каких-либо функций, некий ящик, который принимает в себя данные, обрабатывает их и возвращает обратно. Данные в модель приходят из контроллера, затем эти данные обрабатываются, будь то запросы к базе данных, либо же бизнес вычисления. Данные, пришедшие из контроллера, могут сохраняться в базе данных. После обработки данных моделью, результат работы возвращается обратно в контроллер. В больших приложениях, как правило, имеется много моделей, каждая из которых отвечает за конкретный модуль, будь то создание пользователей или вывод списка записей в таблицу. В модели определяется ограничения по обрабатываемым данным и правила манипуляции данными.

Представление отвечает за вывод данных для пользователя. Обычно представление содержит в себе мало логики и предназначено только для отображения результата работы модели. В современных сайтах большое внимание уделяется клиентской части приложения. Поэтому скрипты в представлениях порой достигают очень больших размеров, что замедляет работу приложения. Представление реагирует на действия пользователя и передает данные в контроллер. Обычно представление реализуется посредством шаблонов html-разметки, которые заполняются данными. Данные приходят из модели, контроллер заполняет представление и шаблон html документа отправляется клиенту.

Контроллер – блок в шаблоне MVC, который получает данные, нормализует их, производит валидацию данных, и после прохождения этих процедур передает данные дальше в нужную модель. С другой стороны, контроллер получает данные из модели, и выбирая нужное представление, отображает их. Контроллер не должен содержать в себе никакой бизнес логики. Вся основная бизнес-логика должна содержаться в слое модели. Контроллер выступает связующим звеном между моделью и представлением.

**1.2.5 Архитектура клиент-сервер**

Обычно компьютеры, входящие в какую-либо систему, не являются равноправными. Так же, как и в социальной группе есть лидер и подчиненные. Некоторые компьютеры несут в себе информацию, а другие же пользуются этой информацией посредством запросов к главному компьютеру.Сервером называется компьютер, который хранит в себе информацию, и который обрабатывает полученные от клиента запросы, а затем выдает ответ. Как правило, на сервере установлено специальное программное обеспечение для выполнения какой-либо сервисной задачи без участия человека. В свою очередь веб-сервером называют как программное обеспечение, которое выполняет функции сервера, так и сам компьютер, на котором это программное обеспечение работает. Клиентом же называют компьютеры, которые посылают запросы на сервер и получают ответ. Клиент и сервер могут находиться как в рамках одной сети, так и в рамках различных сетей. Предполагается, что приложение запущено и работает на сервере.

Суть клиент-серверной архитектуры заключается в разделении приложения на три группы:

- отображение и ввод данных для пользователя;

- функции связывающие отображения и бизнес-логику;

- функции управления ресурсами (работа с базой данных).

Из этого следует, что любое приложение можно разделить на три компонента:

- компонент отображения;

- прикладной компонент;

- компонент управления ресурсами.

Отправка данных от клиента серверу осуществляется с помощью приложения, называемого браузером. Браузер не имеет привязки к операционной системе, поэтому веб-приложения являются кроссплатформенными. Браузер в свою очередь интерпретирует HTML документ своим образом и отображается пользователю. Чтобы браузер мог отображать данные для пользователя, серверу необходимо посылать ответ в формате HTML.

Веб-сервер отвечает за обработку данных, полученных от клиента. Веб-сервер получает данные от клиента, активизируя действие по определенному пути, передает данные в логические модули программы. После этого приложение обрабатывает полученный запрос и веб-сервер формирует HTML страницу, которую отправляет назад клиенту. Сервер содержит ряд шаблонов HTML, на которые накладываются данные и отсылаются клиенту.

Отличительной особенностью веб клиент-серверной архитектуры является то, что в роли клиента выступает браузер, который посылает запросы на сервер, посредством HTTP/HTTPS протоколов. Веб-сервер принимает запросы от клиента, обрабатывает данные, и посылает ответ клиенту. А взаимодействие между клиентом и сервером осуществляется через интернет соединение.

**1.2.6 База данных**

В качестве системы управления базами данных будет использоваться PostgreSQL.

PostgreSQL — свободная объектно-реляционная система управления базами данных (СУБД). PostgreSQL является одной из наиболее популярных систем управления базами данных [!]. Сам проект postgresql эволюционировал из другого проекта, который назывался Ingres. Формально развитие postgresql началось еще в 1986 году. Тогда он назывался POSTGRES. А в 1996 году проект был переименован в PostgreSQL, что отражало больший акцент на SQL. И 8 июля 1996 года состоялся первый релиз продукта. С тех пор вышло множество версий postgresql. Текущей версией является версия 14.2. Однако регулярно также выходят подверсии. PostgreSQL поддерживается для всех основных операционных систем – Windows, Linux, MacOS.

Особенности PostgreSQL:

- высокая производительность;

- надежность;

- гибкость и простота в использовании;

- многоверсионность;

- репликация;

- масштабируемость.

**1.3 Постановка задачи**

Сегодня рынок программного обеспечения выставляет высокие требования ко всем разрабатываемым программным продуктам. Для современных программных средств важными требованиями являются мультиплатформенность, масштабируемость и переносимость. Использование вышеперечисленных технологий при разработке дипломного проекта позволяет сократить время на разработку, увеличить качество кода и засчет этого выполнить данные требования

Объектом исследования является процесс создания объектов и обработки данных, поступающих на сервер. Предметом исследования является создание программного комплекса, обеспечивающего полноценную работу веб-приложения.  
 Основными требованиями, которые были заложены в основу при разработке программного комплекса дипломного проекта стали: простота использования и расширяемость.

На основании вышесказанного, для разработки программного продукта были определены следующие задачи:

- разработка структуры БД;

- разработка серверной части;

- разработка клиентской части.

Веб-приложение будет представлять из себя сайт, который будет предоставлять следующие функции:

- регистрация пользователя в системе;

- изменение данных пользователя;

- создания, редактирование и удаление объявления о продаже;

- отправка письма на электронную почту о получении заказа;

- администрирование пользователей, товаров;