ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»
(НИУ «БелГУ»)

ИНСТИТУТ ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ И ЕСТЕСТВЕННЫХ НАУК

КАФЕДРА МАТЕМАТИЧЕСКОГО И ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННЫХ СИСТЕМ

РЕАЛИЗАЦИЯ И ИССЛЕДОВАНИЕ ИНТЕЛЛЕКТУАЛЬНОГО ПОИСКА И ФИЛЬТРАЦИИ ДАННЫХ В ВЕБ-ПРИЛОЖЕНИЯХ С ИСПОЛЬЗОВАНИЕМ МЕТОДОВ ПРИНЯТИЯ РЕШЕНИЙ И ИХ КОМБИНАЦИЙ

Магистерская диссертация обучающегося по направлению подготовки
02.04.01 Математика и компьютерные науки
очной формы обучения, группы 07001531
Нджугуна Габриел Нджороге

Научный руководитель
доцент кафедры информационно-телекоммуникационных систем и технологий НИУ «БелГУ», к.т.н.,
доцент Девицына С.Н.

Рецензент
доцент кафедры прикладной информатики и информационных технологий НИУ «БелГУ», к.т.н.
Маматов Е.М.

БЕЛГОРОД 2017

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Интернет - глобальное информационное пространство, основанное на самых передовых технологиях, обладающее широким спектром информационных и коммуникационных ресурсов, содержащее колоссальные объемы данных. Развитие Интернета стало причиной возникновения электронного бизнеса (электронной коммерции), социальных сетей, электронных баз данных и хранилищ информации, электронных библиотечных систем. Поиск, систематизация, быстрый обмен и хранение больших объёмов информации являются основными условиями конкурентной борьбы на рынке электронной коммерции.

Для поиска информации в сети Интернет используются поисковые системы, такие, как Google, Яндекс, Yahoo!, Рамблер и другие. Тем не менее, поиск и систематизация информации занимает очень много времени, и классические алгоритмы поиска перестали удовлетворять пользователя.

Поэтому разработка и исследование систем интеллектуального поиска и фильтрации данных в веб-приложениях является актуальной задачей, одно из решений которой представлено в данной магистерской диссертации.

Целью магистерской диссертации является разработка программного обеспечения для создания интеллектуальной поисковой системы с использованием методов принятия решений и их комбинаций.

В магистерской диссертации решены следующие задачи:

- обзор методов поиска информации;
- анализ методов реализации поисковых систем;
- разработка и тестирование ПО для создания поисковой системы.

Объектом исследования является поисковая система, предметом исследования – методы интеллектуального поиска и фильтрации данных в

веб-приложениях на основе алгоритмов профилирования пользователей и механизмов коллаборативной фильтрации.

ГЛАВА 1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Обзор видов поисковых систем и методов поиска

Интернет для пользователя является источником информации, средством обмена информацией и местом размещения личной информации.

В 1992 году была изобретена служба, получившая название «Всемирная паутина» (World Wide Web, или WWW, Web (веб)). WWW позволяет любому пользователю Интернета представлять свою информацию в мультимедийной форме. [1]

Наиболее ощутимое влияние Интернет оказывает на информационно-библиографическую деятельность. Электронные библиотеки являются хранилищами данных, позволяющими в любой момент времени, из любой точки получить необходимую информацию. Тот факт, что энциклопедические, справочные и библиографические источники трансформируются в электронную форму быстрее любых других видов документов, уже в ближайшее время приведет к тому, что цифровые или электронные ресурсы и технологии будут полностью доминировать в информационно-библиографической деятельности библиотек. Согласно данным Интернет-статистики государственной публичной научно-технической библиотеки России (ГПНТБ), спрос на электронные ресурсы в настоящее время в 5-7 раз превосходит спрос на ресурсы печатные. Это означает, что электронные ресурсы «работают» в 50-70 раз активнее, чем печатные.

В отличие от поиска документов в библиотеке или архиве, поиск в Интернете не дает в руки пользователя непосредственно сам ресурс. При таком поиске определяется только место, где ресурс физически хранится. Это место называется адресом ресурса. Пользователю сообщаются все адреса, где находятся ресурсы, которые могут представлять для него интерес. Затем

пользователь сам выбирает потенциально интересные ему адреса. Адрес ресурса называется URL-адрес (Uniform Resource Locator Унифицированный указатель ресурса). [4]

Все поисковые системы объединяет то, что они расположены на специально выделенных мощных серверах и привязаны к эффективным каналам связи. Поисковые системы называют еще информационно-поисковыми системами (ИПС). Количество одновременно обслуживаемых посетителей наиболее популярных систем достигает многих тысяч. Самые известные обслуживают в сутки миллионы клиентов. В случаях, когда поисковая система имеет в своей основе каталог, она называется каталогом. В ее основе лежит работа модераторов. В основе же ИПС с полнотекстовым поиском лежит автоматический сбор информации. Он осуществляется специальными программами. Эти программы периодически исследуют содержимое всех ресурсов Интернета. Для этого они перемещаются, или как говорят, ползают, по разным ресурсам. Соответственно такие программы называются роботы. Есть и другие названия: WWW поскольку - это аббревиатура выражения Всемирная паутина, то такую программу принято назвать «спайдером» от англ. - Паук. В последнее время используются другие названия: автоматические индексы или директории. Все эти программы исследуют и «скачивают» информацию с разных URL-адресов. Программы указанного типа посещают каждый ресурс через определенное время. Ни одна поисковая система не в состоянии проиндексировать весь Интернет. Поэтому БД, в которых собраны адреса проиндексированных ресурсов, у разных поисковых систем разные. Тем не менее, многие из них стремятся, по возможности, охватывать в своей работе все пространство мировой сети. Это универсальные системы.

Автоматические индексы или директории. Все эти программы исследуют и «скачивают» информацию с разных URL-адресов. Программы указанного типа посещают каждый ресурс через определенное время. Ни

одна поисковая система не в состоянии проиндексировать весь Интернет. Поэтому БД, в которых собраны адреса проиндексированных ресурсов, у разных поисковых систем разные. Тем не менее, многие из них стремятся, по возможности, охватывать в своей работе все пространство мировой сети. Это универсальные системы.

Работа поисковой системы обеспечивается тремя составляющими:

- Программа «робот» (спайдер). Она анализирует ресурсы и производит их индексацию.

- Индексы поисковой системы. Они формируют создаваемые поисковой системой собственные БД.

- Программа, которая в соответствии с запросом пользователя готовит ему ответ на основе анализа индексов, то есть собственных БД.

Пользователь реально имеет дело только с последней из этих трех составляющих. Мощные поисковые системы универсального типа созданы для работы на всех основных языках мира. Каждая страна старается создать хотя бы одну собственную поисковую систему.

1.2 Анализ проблем поиска информации в Интернете

В течение последних нескольких лет всемирная паутина стала самым популярным местом для людей, чтобы публиковать и получать информацию. По оценкам, в настоящее время насчитывается более трех миллиардов общедоступных веб-страниц в Интернете. Поскольку все больше и больше веб-страниц добавляются в Интернете, быстрый поиск полезной информации стал проблемой для миллионов веб-пользователей. В настоящее время существует два подхода к поиску нужных данных в Интернете.

Пользователь может начать просматривать из начальной страницы и следовать определенным гиперссылкам для перехода к другим страницам. Активность гиперссылок может повторяться до тех пор, пока пользователь

не нашел нужную информацию. Ключ к успешному просмотру - стартовая страница. Для облегчения просмотра веб-страницы могут быть организованы в иерархии категорий, таких как в Google (www.google.com). В типичной иерархии категорий корень содержит очень общие категории, например, образование и спорт. Каждая категория, в свою очередь, содержит меньше общих категорий. Эта иерархия помогает сузить информационное пространство постепенно и в конечном итоге приводит к желаемой информации.

Пользователь может использовать поисковую систему, чтобы найти нужную информацию. Из интерфейса поисковой системы пользователь отправляет запрос, который описывает его информационные потребности. Когда запрос обрабатывается с помощью поисковой системы, перечень документов, возвращается пользователю, как правило, с лучшими совпадениями, отображается в верхней части. Многие поисковые системы работают в Интернете. Некоторые из наиболее известных - AltaVista, WebCrawler, Hotpots, Google и Lycos. Чтобы включить быструю оценку запросов пользователей, большинство поисковых систем применяет возможность поиска путем создания инвертированного индекса файла для этих документов.

Каждый из указанных выше двух подходов имеет свои преимущества и недостатки по сравнению с другим. Просмотр, как правило, даёт более высокое качество результата (т.е. меньше нежелательных страниц), когда используется хорошая начальная страница. Иерархия категории часто используется для руководства просмотром и может быть создана, в основном, вручную, что требует много времени и сил. В результате лишь очень небольшая часть веб-страницы может быть классифицирована. Кроме того, перемещения по иерархии могут быть утомительными и пользователь может иногда теряться во время процесса из-за нескольких факторов:

- иерархия обычно становится очень сложной, после первых нескольких уровней, и пользователь не может знать, какие суб-иерархии находятся рядом;

- необходимые данные могут быть размещены в различных суб-иерархиях;

- пользователь может запутаться в иерархии, если на пути данные организованы не так, как ожидается пользователем. Это возможно, потому что разные люди применяют различные способы категоризации данных (то есть, категоризация очень субъективна). В отличие от этого, поисковая машина может автоматически индексировать большое количество веб-страниц, и в результате, один запрос может часто приводит к получению большого количества страниц, возвращаемых в ответ.

Основная проблема существующих поисковых систем - пользователи часто перегружены лишней информацией и получают высокий процент нежелательных страниц в результатах поиска. Тем не менее, поисковая система является быстрым способом получения нужной информации в Интернете и миллионы пользователей используют поисковые системы ежедневно.

Поисковая система представляет собой систему поиска текста на веб-страницах. Основные технологии, используемые для построения систем текстового поиска, также используются для построения поисковых систем. Веб-среда, по сравнению с традиционной средой текстового поиска, имеет уникальные характеристики и является причиной разработки новых технологий для поисковых систем.

1.3 Описание структуры системы поиска информации

Поисковые системы разработаны и реализуется по оригинальной архитектуре. Система состоит из компонентов: сканер, поисковик, и базы данных. На рис. 1.1 показана общая структура поисковой системы.

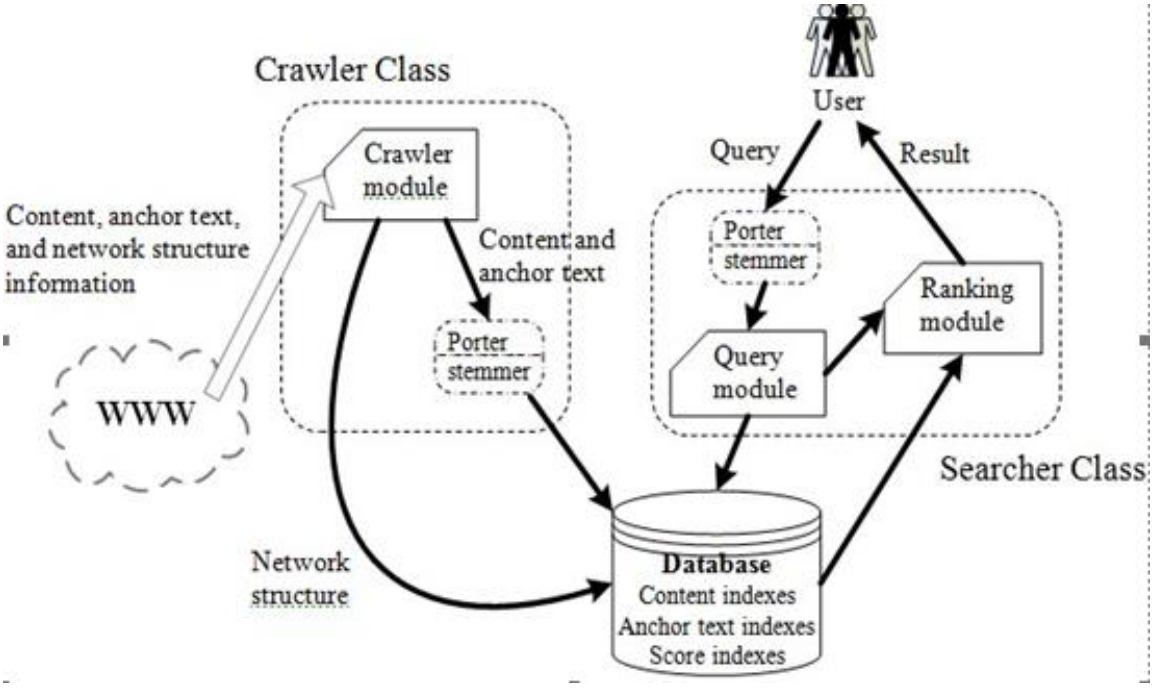


Рис. 1.1. Архитектура поисковой системы высокого уровня

На рис. 1.2 и 1.3 показаны структуры поискового робота (crawler class) и поисковика (searcher class).

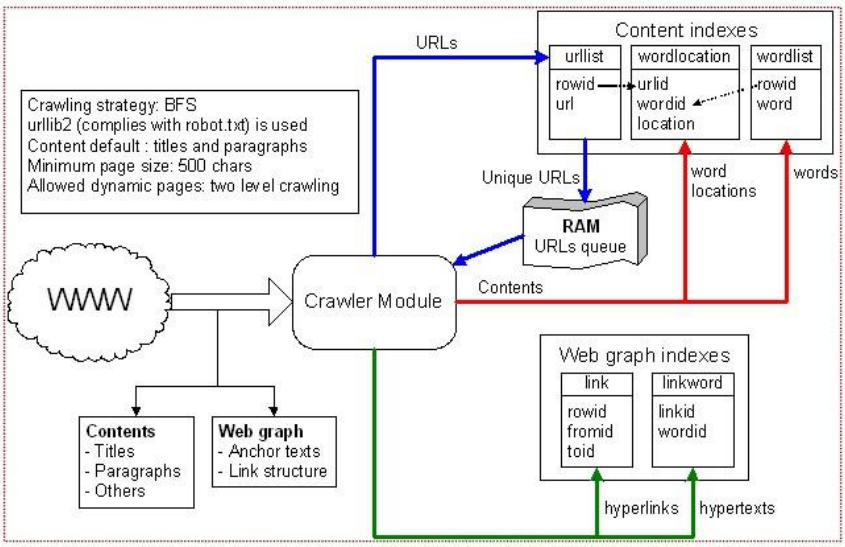


Рис. 1.2. Структура сканера

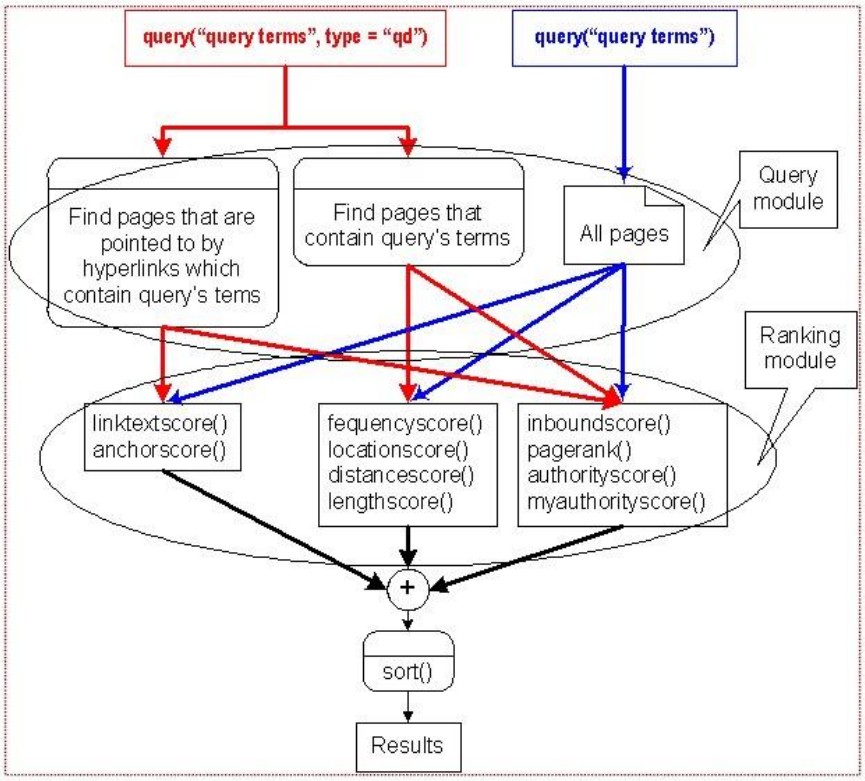


Рис. 1.4. Структура поисковика

Зная исходный код, сканер будет пытаться открыть URL случайным образом в списке URL-адресов, предоставленного пользователем. Если это не удается, сканер начнет со следующего URL. Возникает проблема с этой стратегией, даже при том, что URL-адрес может быть открыт, это не означает, что содержание может быть считано. И далее, если содержание может быть прочитано, иногда функции индексатора в поисковике не может вводить индексы в соответствующие таблицы в базе данных. Во многих случаях эти ошибки могут привести к сбоям в работе, что очень неудобно, так как в веб-данных возможность продолжить процесс обхода путем игнорирования ошибок является необходимым качеством.

Можно преодолеть этот недостаток просто вводя дополнительные попытки, за исключением заявлений для открытия и чтения URL-адресов и целей индексации. На рис. 1.4 показана более подробная информация о конструкции системы.

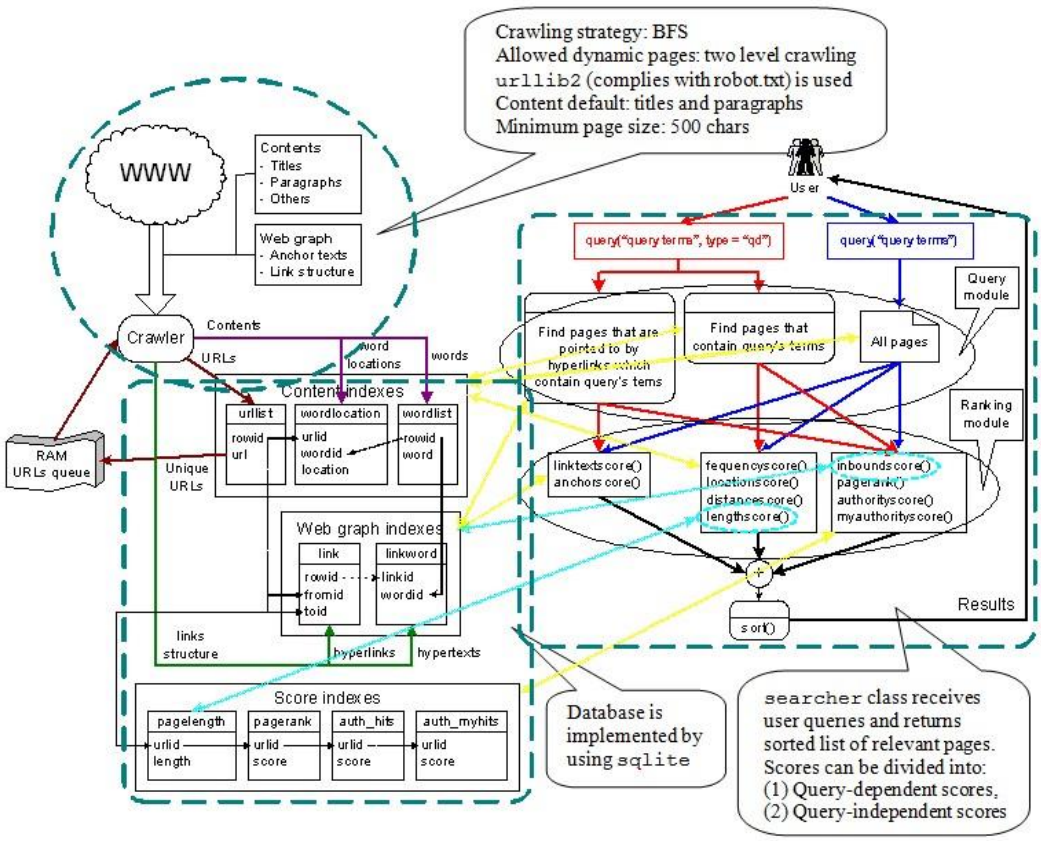


Рис. 1.4. System Design (Системный дизайн)

По сравнению с оригинальной системой, где сканирование закончится сразу, если есть ошибка, это значительное улучшение (финальная версия системы доступна на pythinsearch.googlepages.com и может сканировать веб-страницы в течение двух месяцев непрерывно [17]).

1.4 Исследование существующих методов фильтрации данных в веб- приложениях

Интернет предоставляет пользователям возможность найти различную информацию, и средства поиска облегчают эту задачу, но, тем не менее, классические поисковые системы не всегда удовлетворяют требованиям абонентов, так как практически вся информация, доступная в сети, не содержит правил определения поведения отдельных языковых конструкций, и поэтому ее поиск, соответствующий запросам пользователя, а также интеграция в рамках конкретной предметной области затруднены. Для

обеспечения эффективного поиска, веб-приложение должно четко понимать семантику документов, представленных в сети. В связи с этим, можно наблюдать бурный рост и развитие технологий Semantic Web, происходящий в настоящее время. Консорциумом W3C была разработана концепция, которая базируется на активном использовании метаданных, языке разметки XML, языке RDF (Resource Definition Framework - Среда Описания ресурса). Все предложенные средства позволяют осуществлять обмен данными и их многократное использование.

Системы рекомендаций - это программы, которые пытаются предсказать, какие объекты (книги, фильмы, музыка, веб-сайты) могут понравиться пользователю, имея определенную информацию о его профиле.

Такие программы используются, как правило, в коммерческих целях (в первую очередь, в Интернет-магазинах, либо на специализированных сайтах «по интересам» с целью предложения товаров). С другой стороны, актуальной задачей является интеллектуализация самого процесса поиска в Интернете. Многие пользователи Интернет объективно полагают, что современные возможности поисковых систем не позволят им найти необходимые документы или данные. Для такого мнения пользователей всемирной сети имеются следующие предпосылки:

- взрывной рост объёмов доступных обществу данных вообще (увеличение числа книг, фильмов, новостей, рекламных сообщений и пр.);

- Увеличение объема онлайновых данных;

- Реальный объём информации, окружающей человека, значительно выше того, что он может реально пропустить через себя, чтобы обнаружить необходимую и достаточную.

Системы рекомендаций актуально использовать для интернет-магазинов. Это позволит пользователю рекомендовать популярный, качественный товар который может его заинтересовать, либо при отсутствии

какого либо товара на складе посоветовать ему аналог запрашиваемой продукции. [5]

1.4.1 Анализ методов расчёта рекомендаций

В алгоритмах систем рекомендаций используются определения. Объект - это песня, фильм, товар, т.е. информационный контент. Информационный контент - то, что потребляют пользователи (системы рекомендаций), и им нужно это рекомендовать. Пользователь - это человек, зарегистрированный в системе, он может покупать, слушать, смотреть, оценивать объекты и пользоваться сервисом. Рекомендация - это объект или несколько объектов, которые система рекомендации выдает пользователю. Система рекомендаций позволяет человеку обозначить свои вкусы и возвращает результаты, Интересные для него, базируясь на оценках других пользователей и своих предположениях.

В отличие от поисковых систем, для получения от системы ответа не требуется четкого задания запроса. Вместо этого пользователю предлагается оценить некоторые объекты из коллекции, и на основании этих его оценок и сравнения их с оценками предыдущих пользователей строятся предположения о вкусах пользователя и возвращаются наиболее близкие к ним результаты, формируя для него персонализированную выдачу.

В качестве набора оцениваемых объектов могут, к примеру, выступать: каталог ссылок на веб-сайты, лента новостей, товары в электронном магазине, коллекция книг в библиотеке и т.п. В сферу применения подобных систем входят и ситуации, когда пользователь не ищет информацию по конкретному ключевому слову, а, к примеру, хочет получить список современных статей, по похожих тематике на те, которые он просматривал до этого. В зависимости от того, какие данные используются для расчета рекомендаций, системы делятся на три больших класса:

- Методы коллаборативной фильтрации
- Методы, анализирующие содержимое объектов
- Методы, основанные на знаниях. [5]

1.4.2 Методы коллаборативной фильтрации

Методы коллаборативной фильтрации заключается в том, что каждого пользователя системы просят высказать свое мнение, выраженное в определенном численном значении на некоторой шкале градации относительно предъявляемого ему ряда объектов. Этими объектами могут быть различные потребительские товары, фотографии, статьи, музыкальные произведения, кинофильмы, телепередачи, компьютерные игры и так далее. По мере того как в базе системы коллаборативной фильтрации набирается все больше и больше собранных оценок, происходят следующие важные вещи:

- Система начинает реально понимать, как выглядят собственные предпочтения каждого отдельного пользователя этой системы; - Система начинает объединять пользователей в группы по схожести их интересов и делится персональным составом групп с самими пользователями, входящими в эти группы;

- Система становится способной дать персональную рекомендацию каждому конкретному пользователю в отношении объектов, с которыми он пока не сталкивался.

Если пользователь оценивает объект, но не знает пока чего-то нового, а люди, очень похожие на пользователя по своим оценкам, оценили это новое, то система предложит пользователю это новое, потому что уверена, что он с высокой степенью вероятности оценит это новое для себя так же, как и те, чьи предыдущие предпочтения совпадают с его. Интересы пользователей представлены оценками, которые они дают объектам после просмотра,

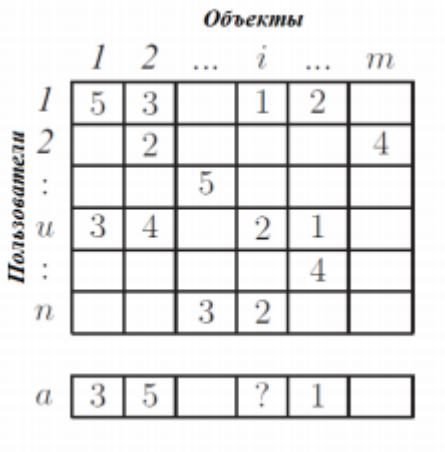
покупки и т.д. Основная идея данных методов заключается в сравнении между собой интересов различных пользователей или объектов на основе этих оценок. При этом никакой дополнительной информации о самих пользователях и объектах не используется. Методы второго класса, наоборот, используют содержимое объектов для получения рекомендаций. Эти методы работают в тех случаях, когда содержимое объектов представлено в виде текстов. Они хорошо подходят для рекомендации книг. Также их можно использовать для сравнения названий, описаний и другой текстовой информации, доступной из фильмов, песен, товаров и т.д. Методы, основанные на знаниях, требуют от пользователя описать свои требования к нужным ему объектам. А затем ищут с использованием своей базы знаний объекты, удовлетворяющие поставленным требованиям.

Для решения задач, поставленных в магистерской диссертации, предлагается использовать метод коллаборативной фильтрации, поэтому рассмотрим этот метод более детально.

Коллаборативная фильтрация (Collaborative фильтрация) - это метод рекомендации, при котором анализируется только реакция пользователей на объекты. Пользователи оставляют в системе оценки объектов. Причем, оценки могут быть как явные (например, оценка по пятибалльной шкале), так и неявные (например, количество просмотров одного ролика).

Конечной целью метода является как можно более точное предсказание оценки, которую поставил бы текущий пользователь системы ранее неоцененным им объектам. Чем больше оценок собирается, тем точнее получаются рекомендации. Получается, пользователи помогают друг другу в фильтрации объектов. Поэтому такой метод называется также совместной фильтрацией.

Пусть в системе есть пользователи и объекты. Пусть некоторые пользователи оценили некоторые объекты. И пусть оценка - это натуральное число от 1 до 5. Тогда все оценки можно изобразить в виде матрицы:



Столбцы в матрице это пользователи, а строки - объекты. Пусть имеется пользователь ***a***. Задача - предсказать, какую оценку поставил бы пользователь ***a*** объекту ***i***. Будем рассматривать только пользователя ***a*** и тех пользователей, которые оценили объект ***i***.

Алгоритм включает в себя 3 шага: Для каждого пользователя U вычислим, насколько его интересы совпадают с интересами пользователя ***a***; После этого выберем множество пользователей, наиболее близких к ***a***; Предскажем оценку на основе оценок объекта ***i*** "соседями" из предыдущего шага.

Первый шаг. Каждому пользователю в матрице R соответствует одна строка. Поэтому будем вычислять близость векторов-строк пользователей. Существует множество способов подсчета близости векторов. Один из самых простых - посчитать косинус между этими векторами:

$$sim(u,a) = \frac{\sum_{i=1}^{m} r_{a,i} r_{u,i}}{\sqrt{\sum_{i=1}^{m} r_{a,i}} \sqrt{\sum_{i=1}^{m} r_{u,i}}}$$

Здесь sim(u,a) - мера близости (похожести) пользователей a и u. $r_{u,i}$ - значение матрицы R: u строка, i столбец. sim(u,a) принимает значения из отрезка [0,1]. Если пользователь не указал оценку для какого-то объекта, соответствующее значение матрицы равно 0. Также часто используется коэффициент корреляции Пирсона, мера близости Дайса, а также другие меры близости.

Второй шаг. Теперь нужно выбрать множество K наиболее похожих на ***a*** пользователей. Есть несколько способов выбора. Чаще всего фиксируется целая константа k. Затем все пользователи сортируются по убыванию меры близости. Во множество K входят первые k пользователей, наиболее близких к ***a***.

Третий шаг. Имея множество K близких пользователей, нужно вычислить оценку, которую поставил бы пользователь ***a*** объекту i. Рассматриваются только те пользователи, которые оценили объект i. Нужная оценка вычисляется по формуле:

$$p_{a,i} = \frac{\sum_{u \in K} r_{u,i} \times sim(a,u)}{\sum_{u \in K} sim(a,u)}$$

Здесь: $p_{a,i}$ - это предсказываемая оценка пользователя ***a*** для объекта i. Она представляет собой среднее по всем пользователям из множества K. Используются веса: чем ближе пользователь u к пользователю a (согласно мере близости sim(a,u), вычисленной на первом шаге), тем сильнее его вклад в предсказание оценки. Таким образом, описанный алгоритм предсказывает оценки для объектов, которые текущий пользователь еще не оценил. Для того, чтобы сделать рекомендацию для данного пользователя, достаточно предсказать оценки для всех неоцененных объектов и выбрать объекты с наибольшей предсказанной оценкой. Для решения поставленной задачи матрица оценок будет не двумерная, а четырехмерная. В ней, кроме объектов и пользователей, будет ещё 2 параметра: время и геолокация. Время будет показывать, какой промежуток времени пользователи тратили на просмотр того или иного объекта, а геолокация будет отвечать за целесообразность заказа товара. [5]

1.5 Обзор существующих методов интернет-маркетинга

Интернет-маркетинг - это комплекс мер, направленных на улучшение ранжирования веб-ресурса в сети, увеличение посещаемости и, как следствие, привлечение новых клиентов и роста компании. Для того, чтобы определить конкретные действия, направленные на улучшение видимости сайта в результатах поиска, используют термины "Поисковая оптимизация" или "продвижение сайта".

Цель интернет-маркетинга - это увеличение эффективности сайта как инструмент современного бизнеса. Основная задача маркетолога - постоянный мониторинг трафика сайта и информации для посетителей, он может получить, рост популярности сайта, мониторинг технических возможностей посетителей, эффективность шагов, предпринятых в области маркетинга. Развитие интернет-маркетинга тесно связано с развитием сети Интернет. Появление глобальных сетей сделало возможными для пользователей покупку, продажу и обмен информацией, распространение рекламных сообщений.

Согласно Википедии [6] в качестве стратегии интернет-маркетинга используется поисковый маркетинг (англ. Search engine marketing (SEM)) – комплекс мероприятий, направленных на увеличение посещаемости сайта его целевой аудиторией с поисковых машин. SEM рассматривает, как работают поисковые системы, что люди ищут, фактические условия поиска или ключевые слова, набранные в поисковых системах и какие поисковые системы предпочитают их целевые аудитории.

SEO - это комплекс мер по оптимизации сайта с целью улучшения его позиций в поисковиках и, как следствие, увеличения его посещаемости. Оптимизация веб-сайта может включать в себя редактирование его содержания, HTML и связанных с ними кодирования для обеих повысит его

значимость для конкретных ключевых слов и устранить препятствия для индексации деятельности поисковых систем.

Продвижение сайта увеличит количество обратных ссылок, или внешних ссылок, что является еще одним инструментом SEO тактики. [3] Google разрабатывает и продвигает мобильный поиск во всех своих продуктах, и многие бренды начинают принимать такой подход к их интернет-стратегии [4].

Методы интернет-маркетинга можно разделить на методы исследования рынка и методы продвижения и продаж. Наиболее эффективные методы современного интернет-маркетинга это:

- поисковая оптимизация;
- Рекламная рассылка;
- контекстная реклама;
- социальные сети;
- видео-маркетинг.

Могут быть использованы следующие методы: прямой сервер записи посетителей, анализ и учет интересов посетителей путем активного взаимодействия с встроенными поисковыми системами, электронные опросы посетителей, интерактивное общение.

1.5.1 Оценка процесса принятия решений

При принятии решения люди используют два типа информации: информация, которую они приобрели через их предыдущий опыт, и дополнительная информация, которую они собирают, чтобы принять решение. Данные из поисковых систем, таких как Google, могут помочь моделировать источники информации.

Статистические данные из поисковых систем на основе контента в Интернете могут помочь оценить предыдущий опыт; и, в частности, такая

статистика может способствовать механизму выбора с отсроченным результатом. Данные поисковых запросов пользователей могут помочь оценить виды интересующей информации, и помочь предсказать последующие решения. Такие данные позволяют сравнить информацию, собранную в разных странах. Данные поисковой системы представляют собой ценный новый ресурс для когнитивистов, предлагая новый инструмент для понимания процесса принятия решений человеком.

Интенсивное взаимодействие с Интернет создаёт цифровое отражение современной человеческой жизни. События по всему миру мгновенно сообщаются через Интернет всем гражданам, и множество дополнительной информации загружается в World Wide Web. Поисковые системы, такие как Google, выступают в качестве шлюза для этого обширного нового информационного ресурса. В процессе создания система делает эту информацию доступной для пользователей, параллельно Google регистрирует, какой контент существует в Интернете, и какую информацию люди ищут.

Статистика запросов данных поисковой системы может проиллюстрировать, как люди принимают решения, предоставляя разработчикам новые методы для моделирования и исследования этих источников информации.

Анализ контента, доступного в Интернете, может помочь моделировать информацию, которую люди ранее приобрели и сохраняют в памяти. Данные о запросах, сделанные в поисковых системах и других онлайновых информационных ресурсах, могут обеспечить новые сведения о том, как люди собирают информацию, прежде чем предпринимать действия в реальном мире, что позволяет сопоставить процессы сбора информации по всему земному шару. Данные поисковой системы представляют собой ценный новый ресурс для исследования процесса принятия решений человеком.

1.5.2 Использование статистики поиска контента в Интернете

Принятие решений человеком зависит от извлечения данных из памяти и процессов поиска информации, которые в основном зависят и отражают статистическую структуру социально-экономической среды [29, 30]. Следовательно, понимание и принятие решений требует, чтобы люди количественно оценивали структуру окружающей среды, для лучшего выбора. При поиске конкретных данных в Интернете, все чаще используют технологию прокси (фильтрации трафика). Одним из источников при формировании баз данных поисковых систем является статистика по частоте поиска определенных документов в Интернете, которая может отражать частоту, с которой люди наблюдают определенные события или значения.

В попытке получить статистику, Оливола и Сагара [27] итеративно искали WEBSITE Архивы Новостей Google, чтобы количественно оценить внимание к определенным событиями, которые люди могли бы фактически наблюдать в заголовках новостей. Данные о внимании средств массовой информации к событию были получены путем поиска в Google Архивы новостей для новостных статей, заголовки которых содержатся ключевые слова, связанные с запросами. Метод опирается на субъективные распределения, которые содержит память людей, и отражают то, что они видят в средствах массовой информации.

Аналогичным образом, данные о распределении частот веб-контента позволяют объяснять и предсказывать временные предпочтения людей [29].

Другие исследования показали, что содержание Интернета может быть использовано для широкого набора переменных, отражающих различные особенности человеческого познания и человеческого опыта. Например, в [35] продемонстрировали тесную взаимосвязь между распространенностью коррупции в географическом местоположении и относительной частоты документов в Интернете, которые связывают это место с коррупцией.

В общем, все большее число исследований показывает, что распределение частот контента в Интернете может также отражать дистрибутивные свойства окружающей среды, а люди подвергаются воздействию, и, как следствие, изменяется содержание человеческой памяти. Таким образом, Интернет располагает полезным, мощным и универсальным инструментом для формирования человеческого опыта и познания.

1.5.3 Использование данных из запросов поисковой системы для изучения процессов сбора информации

Данные из поисковых систем не ограничиваются статистическими данными веб-страниц. Поисковые системы, такие как Google, и интернет-ресурсы, такие как онлайн-энциклопедии (например, Wikipedia), могут регистрировать каждый запрос, который пользователи Интернета по всему миру делают при поиске информации. И Google и Wikipedia делают эти данные доступными для общественности, что позволяет увидеть, как часто люди искали слово "ресторан" или "рецепт" на Google на данной неделе. При поиске информации в Интернете в дальнейшем, чтобы помочь в последующих решениях, например, где они должны питаться вне дома, или что они должны приготовить в домашних условиях, будет возможно связать данные интернет-поиска с последующими реальными решениями, т.е. система «подсказывает» и предвосхищает события, ориентируясь на предыдущий запрос. Таки образом реализуются механизмы персонализации пользователя.

Для решения этого вопроса, требуется набор данных, описывающих решения, принятые в реальном времени. Обширные транзакционные наборы данных из финансовых рынков представляют собой именно такие данные - решения, принятые трейдерами. Такие большие наборы данных из финансовых рынков особенно важны, учитывая их влияние на экономику.

Прейс, Ров, и Стэнли [36] исследовали, может ли быть найдена взаимосвязь между частотой, с которой пользователи Интернета искали финансовые термины на Google и Wikipedia и последующими действиями, предпринимаемыми на финансовых рынках.

Они изучили эту связь путем внедрения гипотетической торговой стратегии, в которой они купили или продали Dow Jones Industrial Average (DJIA), в зависимости от изменений, как часто пользователи Интернета искали данный термин на Google (Прейс и др., 2013) или просматривали данную страницу в Википедии (Ров и др., 2013). Там, где они видели снижение такой онлайн-активности на данной неделе, они купили DJIA в начале следующей недели. С другой стороны, где они увидели увеличение онлайн-активности на данной неделе, они продали DJIA в начале следующей недели. Оба анализа Рвом и др. (2013) и Прейс и др. (2013) предположили, что увеличение поисков в финансовом отношении связанных с ним терминов, как правило, сопровождаться падением цен на фондовом рынке в России. Прейс и др. (2013 г.) далее показали, что слова, которые были классифицированы как более финансово актуальные, на основе частоты их появления в Financial Times, нормированной по частоте в Интернете в целом, были более тесно связаны с последующими движениями фондового рынка. Точно так же, Ров и др. (2013) показали, что увеличение взглядов Wikipedia страниц, касающихся компаний, перечисленных в DJIA, или более общих экономических понятий, как правило, следуют падениям на фондовом рынке.

Отдельное исследование показало, что данные запроса поисковой системы могут быть использованы для сравнения информации для поиска по всему земному шару, с различиями между странами. Прейс, Ров, Стэнли, и Бишоп (2012) использовали данные о поисках в течение многих лет, выраженных арабскими цифрами (например, "2012"). Для 45 стран с более чем 5 миллионов пользователей Интернета в 2010 году, Прейс и др. (2012), который рассчитывается отношение числа запросов Google на предстоящий

год («2011») к числу поисковых запросов за предыдущий год ("2009"), количество, которое они назвали "будущий индекс ориентации." Их анализ показал, что будущий индекс ориентации сильно коррелирует с ВВП на душу населения. В частности, страны с более высоким ВВП на душу населения характеризовались моделью интернет-поиска, которая не была сосредоточена на настоящем или недавнем прошлом, а на ближайшем будущем, и на их ретроспективном временном горизонте.

Рассмотренные результаты исследований позволяют предположить, что данные из Интернета могут дать новое понимание двух важнейших компонентов процесса принятия решений человеком: во-первых, информацию о мире, как наблюдавшуюся ранее и сохранившуюся в памяти; а во-вторых, процесс сбора новой информации для поддержки решения.

Результаты анализа предполагают, что поисковая система и интернет-данные энциклопедий могут внести свой вклад в понимание того, как собирается информация, прежде чем фондовый рынок падает, и позволяют сравнить, как собирается информация по всему миру, анализ показывает больший акцент на будущее и более широкую ретроспективу в странах с более высоким ВВП на душу населения.

1.6 Исследование существующих методов интеллектуального анализа данных

World Wide Web предоставляет огромное количество необходимых данных в цифровом виде, в качестве доступных гипертекстовых данных могут быть веб-страницы, изображения, информация и др. Этот гипертекстовый пул динамически меняется, по этой причине труднее найти полезную информацию. Для автоматического анализа данных является полезным веб-сканер.

Администраторы базы данных, специалисты по управлению или другие лица, желающие осуществлять интеллектуальный анализ данных из большого количества веб-страниц, потребуют услуги поискового робота или инструментов на его основе. По этим причинам поисковые роботы, как правило, многопоточные, с их помощью миллионы веб-страниц могут быть извлечены параллельно только за один процесс.

Поисковый робот применяет распределенный подход, включая интеллектуальный анализ данных. Он работает, как распределенная сетевая система с блоком управления и возможностью предоставления рабочих мест на другом компьютере, которые связаны с сетью. Процедура позволяет распределять вычислительную мощность, используя различное количество компьютеров, систем, связанных друг с другом. [6]

1.6.1 Представление каталога

Это важный метод в поисковой оптимизации, чтобы создать ближайшие ссылки на веб-сайт через индексируемые страницы и категории. Каталог предоставляет бесплатный сервис для веб-сайта. Запрос информации у Справочника представления проводится по URL, названию, ключевым словам.

Usability Tools - это инструменты для отображения страниц с разным разрешением, другой операционной системы и другого браузера. К ним относятся HTML и CSS проверки, расширение Firefox, а также тест на скорость страницы.

1.6.2 Генерация ключевых слов

Для поисковой оптимизации нужны слова, чтобы уточнить информацию, основанную на этих словах. Ключевые слова должны соответствовать предмету поиска. Этот процесс может реализовываться

различными онлайн-инструментами, как трекеры, Yahoo-инструмент селектора ключевого слова, ключевые слова Google AdWords.

Инструмент ключевых слов (Keyword Tool) - использует функцию автозаполнения (поисковых подсказок). Генерируемые ключевые слова будут использовать выбранный вами язык и домен Google.

1.6.3 Обмен ссылками

Для запуска любого веб-сайта для нужд бизнеса, нужно организовать взаимный обмен ссылками с другими сайтами. Это процедура предусматривает ссылку на другой сайт и другие веб-сайты, которые размещают на сайте. Link Tool (Инструмент Ссылка) - эти инструменты включают в себя ссылку на популярные ресурсы, с помощью которых ранжирование сайта может быть увеличено.

1.6.4 Методы сканирования

1) Focused Crawling - целенаправленный сканер, предназначен только для получения документов по определенной теме, тем самым уменьшая объем сетевого трафика и загрузки. Цель состоит в том, чтобы выборочно искать страницы, которые имеют отношение к заранее определенному набору тем. Это приводит к экономии аппаратных средств и сетевых ресурсов и помогает поддержать поиск в актуальном состоянии.

2) Метод распределенного сканирования - распределение активности через несколько процессов, этот метод позволяет построить масштабируемую систему, которая является отказоустойчивой. Распределяя нагрузку, снижает требования к оборудованию, и в то же время увеличивает общую скорость загрузки и надежность.

Вывод к разделу

Поиск и систематизация информации занимает очень много времени, и классические алгоритмы поиска перестали удовлетворять пользователя. Поэтому разработка и исследование систем интеллектуального поиска и фильтрации данных в веб-приложениях является актуальной задачей. Для решения задач, поставленных в магистерской диссертации, предлагается использовать метод коллаборативной фильтрации.

ГЛАВА 2 ПОСТАНОВКА ЗАДАЧИ СОЗДАНИЯ ПРОГРАММНОГО ПРИЛОЖЕНИЯ

2.1 Применение методов интеллектуального поиска и фильтрации данных в веб-приложениях в современных поисковых системах

Веб-поиск считается дополнительным методом поиска для систематического обзора. Основной метод поиска обычно состоит из библиографического поиска в базе данных, которая используется для получения журнальных статей, тезисов докладов конференции и т.д. Есть исследования, показывающие, что библиографические базы данных могут быть адекватно заменены веб-поисковой системой [1].

В этой главе проведен анализ работы поисковых систем и пути решения проблем, с которыми сталкиваются пользователи. Проанализируем особенности технологии веб-сервера, баз данных и на стороне сервера, которые могут потребоваться для разработки надлежащего решения проблемы оптимизации поиска. Рассмотрим преимущества и недостатки каждой технологии для решения проблемы и предоставим обоснование выбранных технологий.

Существуют тысячи поисковых систем в современном мире, которые обеспечивают легкость поиска любых видов продуктов, находящихся в Интернете. Люди делают заказ и приобретают множество товаров в Интернете, или продают товары в сети. Эти операции реализуются через поисковую систему и делают жизнь проще и комфортнее, так как все может быть реализовано, не выходя из собственной зоны комфорта.

Google или любая другая поисковая система должна обеспечить достаточную скорость поиска. Использование автозаполнения позволяет реализовать данное требование. Стандартная стартовая страница Google будет отображать выпадающий список предложений, поставляемых с

помощью поисковой системы Google. Эта опция может быть удобным способом, чтобы обнаружить подобные, связанные поиски.

2.2 Удобство и простота

Якоб Нильсен, эксперт в области веб-юзабилити, определяет юзабилити как «атрибут качества, который оценивает, насколько легко пользовательские интерфейсы для использования». [1] Слово «юзабилити» также относится к способам улучшения легкости в использовании в процессе проектирования. Важность юзабилити не следует недооценивать, так как удобство использования становится аспектом конкурентной бизнес-среды. Если пользователь не сможет эффективно использовать сайт, то прекратит его использование. Есть много структур и руководств обеспечения юзабилити, доступных на сегодняшний день, но в диссертации будут рассмотрены два: контрольный список [2] и контрольный перечень общего назначения [3].

2.2.1 Структура «Контрольный список»

Первая структура - это веб-сайт юзабилити «Контрольный список», созданный Л. Томасон [2]. Контрольный список выдвигает на первый план 5 важных компонентов полезного веб-сайта: Дизайн с четкой и простой системой навигации; обеспечение ясного и простого содержания; поддержка своего бренда; обеспечение обратной связи для посетителей; и тестирование сайта на реальных пользователях. Этот контрольный список ясен и прост, но дает точные детали о том, что используемый веб-сайт должен и не должен включать в себя.

2.2.2 Структура «Контрольный перечень общего назначения»

Бринк, Gergle и Вуд [3] выделили 10 руководящих принципов, которые состоят из принципов общего назначения для быстрого просмотра веб-сайта. Эти рекомендации, однако, расплывчаты, и для того, чтобы эффективно использовать их, разработчики должны быть знакомы с подробными руководящими принципами. Они предоставляют подробный перечень общих вопросов юзабилити, как правило, используемый при оценке системы. Этот контрольный список проверки предназначен для очень специфических аспектов юзабилити, и требует ответа «да или нет».

2.2.3 Использование фреймворков

Согласно Википедии, компьютерная база представляет собой «согласованный набор структурных компонентов программного обеспечения, который служит для создания основы, а также основные направления всех или части программного обеспечения». Другими словами, однородная база с готовыми блоками. Есть готовая структура (фреймворк) для всех языков программирования, например, PHP. Язык программирования PHP, особенно в связке с базой данных MySQL - оптимальный вариант для создания веб-сайтов различной сложности.

Полезность фреймворка заключается в том, что возможно избежать расходов времени на разработку того, что уже сделано другими, часто более опытными, и который в дополнение были использованы и подтверждены многими пользователями. Можно представить структуру, как набор доступных инструментов. Например, разработчик должен сделать маршрутизацию для сайта, он берёт уже готовый компонент и использует его. Это приводит к экономии времени, надежности, обновления в случае необходимости.

Laravel. Laravel - бесплатный веб-фреймворк с открытым кодом, предназначенный для разработки с использованием архитектурной модели MVC (англ. Model View Controller - модель-представление-контроллер). Laravel, созданный Taylor Otwel, инициирует новый способ проектирования структур, используя то, что лучше всего подходит для каждой функции. Например, любое веб-приложение, необходимое системе, которая обрабатывает HTTP-запросы. Вместо того, чтобы изобрести что-нибудь, дизайнер Laravel просто использовал один из наборов автономных компонентов Symfony для создания эффективной системы маршрутизации. Кроме того, отправка электронной почты осуществляется с помощью библиотеки Swift Mailer.

2.2.4 Выбор структуры юзабилити

В диссертации решено использовать оба контрольных списка. Структура 1 будет использована, чтобы реализовать приложение, поскольку она подробна и дает точную информацию о том, что используемый веб-сайт должен и не должен включать в себя. Будет использована структура 2 в качестве контрольного списка оценки для проведения проверки юзабилити создаваемого в диссертации приложения на стадии тестирования, а также как основа для общей реализации и построения приложения поисковой системы вместе с пользовательским интерфейсом.

2.3 Пользовательские интерфейсы

Пользовательский интерфейс предназначен для взаимодействия пользователя с системой. Он должен скрыть сложность системы, позволяющей пользователю легко и эффективно управлять системой. Интерфейс для веб-приложения работает точно таким же образом. Holzchlag

[5] классифицирует 5 особенностей дизайна интерфейса. Это метафора; ясность; консистенция; ориентация; и навигация. Метафора относится к символическому представлению областей сайта, например, используя знакомые образы для точек входа, выхода и окна в среде. Ясность требует, чтобы каждая активная область на сайте должна иметь причину быть там, и что эта причина должна быть очевидной для пользователя. Например, изображения и кнопки должны выполнять свои функции. Третья особенность, чтобы рассмотреть последовательность, в которой говорится, что использование метафор и навигационных средств равномерно используется. Ориентация означает, что пользователь должен точно знать, где он находится на каждом шагу с помощью приложения. Средства для обеспечения этого включают в себя заголовки, колонтитулы и т.д. Последняя особенность выделенная Holzchlag [5], является важность хорошей навигации, которая может совпадать с вопросами дизайна макета.

2.4 Удобство технического обслуживания

Традиционно требуются знания и опыт веб-дизайнеров для управления контентом. Веб-контент становится более динамичным и основывается на базах данных, позволяя изменения контента производить путем простого изменения информации в текстовых файлах или базах данных, а изменения сразу становятся очевидными в веб-страницах. Термин часто используется для обозначения систем, которые делают управление содержанием проще, такими системами являются системы управления контентом.

2.4.1 Системы управления контентом (CMS)

Рассел Накано [6] показывает CMS, как «информационную систему или компьютерную программу, которую можно использовать для обеспечения и организации совместного процесса создания, редактирования

и управления контентом». CMS позволяет в пределах организации обновить веб-страницы без каких-либо предварительных технических знаний, что является идеальным решением проблемы многих пользователей. CMS поэтому описывается как система контент-менеджмента, с набором макетов дизайна и содержимого, предоставляющего доступ к БД. Возможно обновление контента, хранящегося в базе данных, что автоматически изменяет содержимое веб-страницы, не влияя на дизайн-макет веб-страницы.

Самый простой способ обновить содержимое в базе данных за счет использования форм, которые могут быть доступны в браузере, что позволяет владельцу сайта сделать маленькие или большие изменения быстро через веб-интерфейс. Это устраняет необходимость иметь в веб-странице пакетов-редакторов, и устраняет необходимость какого-либо программирования пользователем. В связи с этим, предлагаемое веб-приложение будет использовать веб-интерфейс, чтобы позволить владельцу поддерживать веб-сайт. Термин «система управления контентом» или CMS, может быть использован для обозначения приложения в остальной части диссертации.

2.4.2 WYSIWYG Редакторы

Редакторы позволяют создавать страницы без знания HTML. Они позволяют использовать макеты текста, изображений и таблиц и т.д., также, как генерируется в любом программном обеспечении для обработки текстов и HTML. Преимуществом этих редакторов является то, что их юзабилити соответствует любым пользователям. [5].

В разрабатываемом в диссертации приложении будет использоваться программное обеспечение - текстовый редактор PhpStorm. Этот редактор является мощным инструментом для создания и редактирования исходного кода. Как и любой другой редактор IDE, он поддерживает базовые функции,

такие как закладки, контрольные точки, подсветка синтаксиса, завершение кода, изменение масштаба, складывание блоков кода, и т.д. Существует множество дополнительных функций, таких как макросы, выделенные TODO элементы, анализ кода, намерение действия, умная и быстрая навигация, и многое другое.

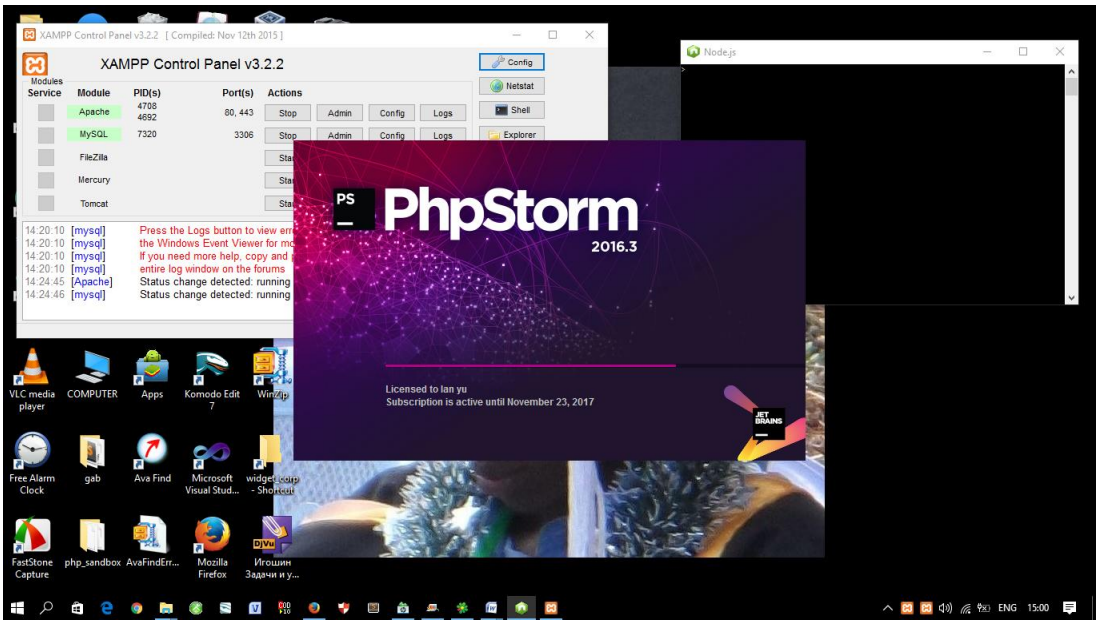


Рис. 2.1 PhpStorm

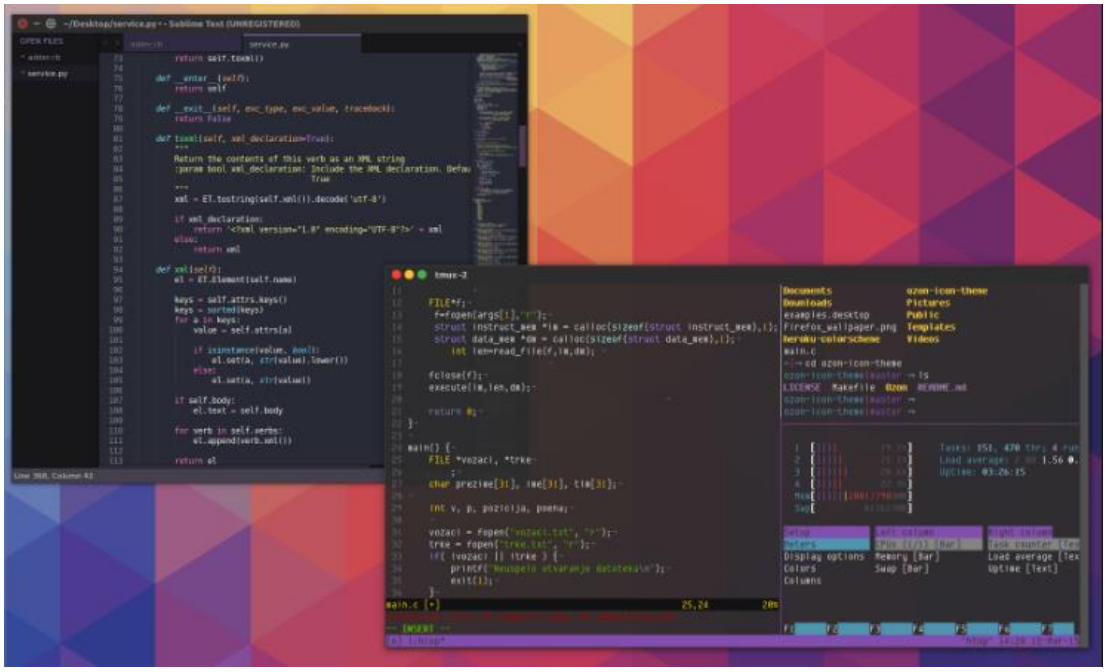


Рис. 2.2 PhpStorm интерфейс

Чтобы настроить среду редактирования, нужно использовать страницу параметров редактора и его дочерних страниц. Существует также команда быстрого переключения, схема, которая позволяет изменять цветовые схемы, темы, ключевые карты и т.д. с парой нажатий клавиш.

Редактор закладок. Все операции с закладками редактора доступны из контекстного меню вкладки, или из окна «Редактор вкладок главного меню».

2.5 Исследование доступных технологий

2.5.1 Веб-сервер

Основная функция веб-сервера - это хранение, обработка и доставка веб-страниц для клиентов. Связь между клиентом и сервером происходит с помощью протокола передачи гипертекста (HTTP). Страницы - HTML документы, которые могут включать в себя изображения, таблицы стилей и скрипты в дополнение к текстовому наполнению.

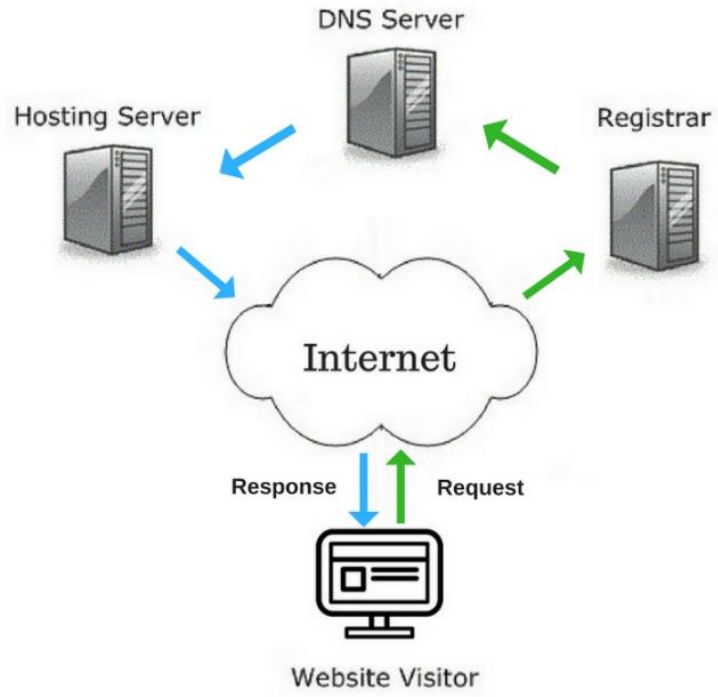


Рис. 2.3. Технологии веб-сервера

Агент пользователя, как правило, веб-браузер или веб-обходчик, инициирует связь, сделав запрос на определенный ресурс, используя HTTP, и сервер отвечает с содержанием данного ресурса или сообщением об ошибке, если не в состоянии сделать это. Ресурс обычно представляет собой реальный файл на вторичном хранилище сервера, но это не обязательно так и зависит от того, как реализован веб-сервер.

В то время как основная функция заключается в обслуживании контента, полная реализация HTTP также включает в себя способы получения контента от клиентов. Эта функция используется для представления веб-форм, в том числе загрузки файлов. Есть два основных типа веб-серверов, обычно использующие веб-хосты. Эти веб-серверы Apache, работающие на веб-серверах UNIX и IIS, работающих на Windows. Каждый веб-сервер будет отличаться в скриптовых технологиях, которые он поддерживает, так как оба сервера должны быть сопоставлены, чтобы определить, какая платформа, предложенная CMS, будет работать дальше.

Самым популярным веб-сервером в Интернете является Apache. Он был самым популярным с апреля 1996 и согласно результатам опроса Netcraft [7], проведенной в ноябре 2005 года "более 70% сайтов в Интернете используют Apache, что делает его более широкое применение, чем все другие веб-серверы вместе взятые". Apache HTTP Server является проектом Apache Software Foundation и всегда активно развивается, в том, что те, кто использует это программное обеспечение может внести свой вклад обратно к нему в виде улучшений и исправлений ошибок. Кроме того, Apache является более быстрым и более стабильным, чем многие другие веб-серверы. Он также может работать на Windows NT / 9x, Netware 5.x и выше, OS / 2 и большинство версий UNIX.

Apache поддерживает PHP (Hypertext Processor Pre), Perl и JavaScript языки, таким образом, обеспечивает доступ к базам данных, таким как

MySQL и MS SQL Server. Это, однако, не поддерживает ASP (Active Server Pages).

Популярность и широкое использование Apache делают его чрезвычайно привлекательным вариантом, так как важно, чтобы разрабатываемое в диссертации веб-приложение содержало меньшее количество ограничений.

2.5.2 Технологии на стороне клиента и на стороне сервера

Pre Hypertext Processor (PHP)

PHP - это язык сценариев на стороне сервера, который предназначен в основном для веб-разработки, но также используется в качестве языка программирования общего назначения. PHP является свободным языком сценариев, который может работать в обоих веб-серверах Apache и МКС. Проблема заключается в том, что не все веб-хосты в настоящее время обеспечивают поддержку для него на своих серверах. [8].

Использовать PHP полезно для добавления интерактивных функций для веб-приложений, т.к. он имеет встроенные объекты в пределах языка, что делает его чрезвычайно совместимым со многими типами баз данных, включая MySQL, MS SQL, Dbase и Oracle. Поэтому его совместимость с базами данных делает PHP главным кандидатом, который будет использоваться в разрабатываемом в диссертации веб-приложении. Несмотря на то, что он не является языком общего назначения, как Java или Perl, PHP относительно легко использовать, потому что он похож на язык C [9].

PHP используется в HTML и потому, что он обрабатывается на стороне сервера, а не на стороне клиента, код PHP скрыт от клиента и не может быть просмотрен путем просмотра исходного кода веб-страницы

Существуют уязвимости в системе безопасности, которые могут быть использованы недобросовестными клиентами, чтобы обойти проверку

безопасности и авторизации. Это часто возникает из-за плохого программирования и данную проблему необходимо исследовать дополнительно, если планируется использовать описанную технологию.

Active Server Pages (ASP)

ASP (активные серверные страницы) являются веб-страницами, которые сочетают в себе HTML и сценарии на стороне сервера. Как и PHP, скрипты сервера обрабатываются на веб-сервере перед отправкой в браузер клиента, что позволяет клиенту увидеть HTML-код. ASP позволяет использовать возможности веб-сервера для обработки запросов пользователей и обеспечить динамическое, индивидуализированное содержание, основанное на логике файлов и баз данных [10]. В отличие от PHP, ASP не является независимым от платформы и требует информационного сервера Microsoft или Personal Web Server для работы, и, следовательно, не поддерживается Apache. Это зависит от языка, и поддерживает использование любого языка сценариев, который совместим с Microsoft Scripting. Например, VB Script, JavaScript и PerlScript.

Объекты данных ActiveX (ADOS) позволяют с легкостью добраться в текстовые файлы и базы данных для управления и отображения информации на веб-странице. Эти компоненты делают ASP совместимым с любыми базами данных, которые работают на операционных системах Microsoft. Кроме того, ASP являются независимыми от типа браузера, делая их доступными во всех браузерах.

Perl

Perl - это высокоуровневый интерпретируемый динамический язык программирования общего назначения, универсальный скриптовый язык. Обычно используется для создания CGI-скриптов для обработки данных HTML-форм на веб-сайте. Одним из факторов, который привел к

популярности языка Perl, является наличие библиотек, которые можно было легко писать программы CGI [11]. Он отличается от PHP и ASP тем, что его код не встроен в HTML. и вместо того, чтобы быть загруженным браузером, находится исключительно на стороне сервера.

Perl был впервые создан Ларри Кингом для программирования с операционной системой UNIX, но благодаря способности обработки больших объемов текста, и его использование в CGI-сценариях, он теперь используется для администрирования баз данных. Perl с помощью дополнительного модуля DBI (Database Interface) используется для упрощения процесса подключения и взаимодействия с базами данных.

В то время как Perl широко используется, я не обладают знанием языка, и, хотя он похож на язык C, к которым у меня есть предварительные знания, оказалось бы очень трудно использовать этот язык.

JavaScript

JavaScript является самым популярным на стороне клиента языком сценариев, и как все языки сценариев может быть встроен в HTML. Это объектно-ориентированный язык программирования, который свободно доступен и может быть использован для добавления интерактивности на веб-страницах. Он является открытым исходным кодом и кросс системная платформа поддерживается обоими серверами Windows, и Linux, что делает его реальным выбором для данного проекта.

Вместо того, чтобы скрипт обрабатывать на сервере, вся обработка осуществляется в браузере клиента. Все основные браузеры, такие как IE, Mozilla, Firefox, Netscape и Opera будут иметь возможность обрабатывать эти сценарии без необходимости применения JavaScript. JavaScript также может определить, какой браузер пользователь использует и загрузить страницу, разработанную специально для этого браузера. Это выгодно, поскольку это

обеспечит членов всех LUUMAFC и сторонников просматривать лучшую возможную версию сайта.

Основными недостатками JavaScript является то, что он требует API (интерфейс прикладного программирования). Чтобы подключить программы Java к базе данных, может вызвать проблемы реализации.

XAMPP

XAMPP является кросс-платформенным веб-серверным пакетом стека с открытым исходным кодом, разработанным Apache Friends, [2], состоящим в основном из базы данных Apache HTTP Server, MySQL, а также переводчиков для скриптов, написанных на PHP и Perl языков программирования. [3,4]. XAMPP расшифровывается как кросс-платформа (X), Apache (A), MariaDB (M), PHP (P) и Perl (P). Это простое, легкое распределение Apache, что делает его чрезвычайно легким для разработчиков, чтобы создать локальный веб-сервер для тестирования и развертывания целей.

Все, что нужно, чтобы создать веб-сервер - сервер приложений (Apache), базы данных (MariaDB), и язык сценариев (PHP). XAMPP также кросс-платформенный, что означает, что он одинаково хорошо работает на Linux, Mac и Windows. Так как большинство веб-серверов используют те же компоненты, как XAMPP, он делает переход от локального сервера тест на реальном сервере очень легко.

Дизайнеры XAMPP намеревались его использовать только в качестве инструмента разработки, чтобы позволить дизайнерам и программистам веб-сайтов проверить их работу на своих компьютерах без доступа к Интернету. Для того, чтобы сделать это как можно проще, многие важные функции безопасности по умолчанию отключены. [11] XAMPP имеет возможность обслуживать веб-страницы на World Wide Web. [12] Специальный инструмент предусмотрен, чтобы защитить паролем наиболее важные части пакета. [13]

XAMPP состоят из следующих компонентов, которые помогают в разработке веб-приложений (рис.2.4).

Component	On Windows	On Linux
Apache 2.4.23	Yes	Yes
MariaDB 10.1.19	Yes	Yes
PHP	Yes - 7.0.13	Yes - 7.0.13[15]
phpMyAdmin	Yes - 4.5.1	Yes - 4.5.2
OpenSSL	Yes - 1.0.2j	Yes - 1.0.2j
XAMPP Control Panel 3.2.2	Yes	No
Webalizer	Yes - 2.23-04	Yes - 2.23-05
Mercury Mail	Yes	No
Transport System 4.63	Yes	No
Tomcat 7.0.56 (with mod_proxy_ajp as connector)	Yes	No
Strawberry Perl 7.0.56 Portable	Yes	No
FileZilla FTP Server 0.9.41	Yes	No
ProFTPD 1.3.4c	No	Yes
GD 2.0.35	No	Yes
Perl 5.16.3	No	Yes
Freetype2 2.4.8	No	Yes

Рис 2.4. Основные компоненты XAMPP

XAMPP также обеспечивает поддержку для создания и управления базами данных в MariaDB и SQLite среди других. После установки XAMPP, можно лечить Localhost как удаленный хост при подключении с помощью клиента FTP. Использование программы, как FileZilla, имеет множество преимуществ при установке системы управления контентом (CMS), как Joomla или WordPress. Кроме того, можно подключиться к Localhost через FTP с HTML-редактором.

Node.js

Node.js является открытым исходным кодом, кросс-платформенная среда исполнения JavaScript для разработки разнообразных инструментов и приложений. Хотя Node.js не является основой JavaScript, [4] многие из его основных модулей написаны на JavaScript, и разработчики могут создавать новые модули в JavaScript. Среда выполнения интерпретирует JavaScript, используя движок V8 JavaScript Google.

Node.js имеет архитектуру, управляемую событиями, способную реализовать асинхронный ввод / вывод. Эти конструктивные варианты

направлены для оптимизации пропускной способности и масштабируемости в веб-приложениях с большим количеством операций ввода / вывода, а также в режиме реального времени веб-приложений (например, в режиме реального времени коммуникационных программ и браузерных игр). [5, 7]

Корпоративные пользователи программного обеспечения Node.js включают GoDaddy, [8] Groupon, [9] IBM, [10] LinkedIn, [11, 12] Microsoft, [13, 14] Netflix, [15] PayPal, [16, 17], SAP, [18] Voxer, [19] Walmart, [20] Yahoo!, [21] и Cisco Systems. Node.js позволяет создавать веб-серверы и сетевые инструменты, используя JavaScript и коллекцию "модулей", которые обрабатывают различные функциональные возможности ядра. [26, 29, 43- 45] Модули предназначены для файловой системы ввода / вывода, сетевые (DNS, HTTP, TCP, TLS / SSL, или UDP), двоичные данные (буферы), функции шифрования, потоки данных [46] и другие основные функции. Модули Node.js используют в API, предназначенных для уменьшения сложности написания серверных приложений. [29, 44]

Оценка клиентских и серверных технологий

Трудности использования JavaScript для подключения к базе данных, предполагают оставить Perl, PHP и ASP в качестве остальных возможностей. PHP и ASP будут использоваться в магистерской диссертации для создания приложения, а Perl решили не использовать, как язык сценариев. Оба PHP и ASP кажутся одинаковыми с точки зрения совместимости баз данных и простоты использования, но в ходе реализации решения приняты меры учесть недостатки безопасности PHPs, и его кросс-платформенная совместимость даёт ему конкурентное преимущество по сравнению с ASP.

2.5.3 Технологии Баз данных

Базы данных MySQL

MySQL является свободным программным обеспечением с открытым исходным кодом, широко доступным и с хорошей технической поддержкой. Он совместим со многими операционными системами, включая Windows, Linux и Mac и может быть использован с веб-серверами Apache и Windows. Он также может обрабатывать большие объемы данных и обеспечивает безопасность посредством авторизации пользователя и привилегии доступа.

MySQL прекрасно сливается с PHP, Perl и ASP, что делает его идеальным для управления контентом веб-приложения. Таким образом, эти стороны сервера могут быть использованы как сценарий для взаимодействия с базой данных MySQL для извлечения и добавления информации в базу данных.

MySQL является центральным компонентом LAMP с открытым исходным кодом стека веб-приложений программного обеспечения (и других стеков "AMP"). LAMP является аббревиатурой "Linux, Apache, MySQL, Perl / PHP / Python". Приложения, которые используют базу данных MySQL включают в себя: TYPO3, MODx, Joomla, Word Press, PHPBB, MyBB, и Drupal. MySQL также используется во многих крупномасштабных веб-сайтах, в том числе Google [10, 11] (хотя не для поисков), Facebook, [12- 14] Twitter, [15] Flickr, [16] и YouTube. [17]

Базы данных Microsoft Access

Microsoft Access является очень популярным приложением для управления данными. Это приложение не с открытым исходным кодом, но доступно через программный пакет Microsoft Office, который является в настоящее время стандартным на всех ПК под управлением операционной системы Windows. Операционные системы Mac и Linux не будут, однако,

иметь доступ к Microsoft Access. Поэтому LUUMAFC в состоянии использовать операционную систему сервера IIS и окна для размещения и редактирования веб-сайта. MySQL с другой стороны, легко интегрируется с Apache и другими веб-серверами, использующими большое количество языков, включая PHP, ASP, Perl и Java. Веб-возможности доступа также ограничены через Интернет, и с MySQL становятся гораздо доступнее.

Доступ и меню являются легкими для неспециалистов, так что пользователь может легко взаимодействовать, не зная Visual Basic. Он также позволяет пользователю создавать ряд форм, чтобы действовать в качестве интерфейса и использовать запросы для взаимодействия с базой данных, что делает его проще в использовании, чем MySQL и делает его идеальным для небольших приложений баз данных. Также может быть использован в качестве интерфейса для доступа к информации о других базах данных, таких как MySQL. Это может оказаться полезным для нетехнических пользователей, незнакомых с SQL и желающих получить доступ к информации непосредственно из базы данных [12].

PostgreSQL

PostgreSQL является открытым исходным кодом, не зависит от платформы системы управления базами данных и похож на MySQL, но менее широко используется. Он предназначен для больших баз данных и включает в себя расширенные возможности по сравнению MySQL [13]. Недостатком PostgreSQL является то, что это происходит медленнее, чем MySQL для выполнения запросов.

MS SQL Server

MS SQL Server является системой управления реляционными базами данных, платформой, которая работает только на серверах Microsoft. Он предназначен для больших баз данных и обрабатывает несколько

пользователей и запросов к базе данных. Он также легко поддерживается с помощью диспетчера GUI Enterprise. Недостатком MS SQL является то, что он не является открытым исходным кодом и дорогим, чтобы купить.

Оценка технологий баз данных

Access предоставляет полезные графические интерфейсы для нетехнических пользователей, веб-приложение может быть получено, что не требует от пользователя изменить базу данных непосредственно. Конструкция должна обеспечить изменение базы данных только через веб-интерфейс.

MS SQL Sever будет полезным СУБД для этого решения, потому что это хорошо при обработке нескольких запросов к базе данных от многократного использования, которые могут только повысить производительность приложения. Основным недостатком MS SQL является его стоимость.

PostgreSQL и MySQL являются наиболее жизнеспособными решениями, поскольку они являются свободно доступными и не зависят от платформы.

ГЛАВА 3. ЭТАПЫ РАЗРАБОТКИ ПРИЛОЖЕНИЯ ДЛЯ ИНТЕЛЛЕКТУАЛЬНОГО ПОИСКА И ФИЛЬТРАЦИИ ДАННЫХ В ВЕБ-ПРИЛОЖЕНИЯХ

3.1 Выбор и сравнение методов

В диссертации представлено создание поискового устройства "Search.dev" - поисковая система, прототип крупномасштабной поисковой системы, что усложняет использование в гипертекстовых структурах. Модели для создания поисковой системы разделены на 2 основные группы; использование иерархической модели; использование гипертекстовой модели. Использование иерархической модели подразумевает многоуровневую рубрикацию информационных ресурсов [10]. Для выбора пути к нужному документу используются описания, составленные службой поддержки данной системы. Гипертекстовая модель позволяет связывать документы ссылками, которые располагаются непосредственно в тексте. Этот проект опирается на ссылки и ключевые слова для поиска информации и это достигнуто через сканер.

К первому этапу работы поисковой системы можно отнести сканирование сайтов в глобальной сети и сбор на свои собственные серверы копий веб страниц. Это образует огромное количество пока ещё не обработанной и не пригодной информации для поисковой выдачи.

Второй этап работы поисковика сводится к приведению в порядок полученной ранее, на первом этапе информации от сайтов. Производится такая сортировка, которая за наименьшее время будет благоприятствовать тому самому качественному поиску, которого собственно и ждут пользователи от поисковой системы. Этап называют индексацией, это значит, что страницы уже являются подготовленными к выдаче, а актуальная база будет считаться индексом.

Третий этап обуславливает поисковую выдачу, после приёма запроса от своего клиента, опираясь на ключевые или около ключевые слова, указанные в запросе. Это способствует отбору наиболее соответствующей запросу информации, и последующей её выдачи. Так как информации очень много, поисковая система выполняет ранжирование в соответствие со своими алгоритмами.

3.1.1 Жизненный цикл разработки системы поиска

Методологии развития жизненного цикла системы (SDLC) – это механизмы для обеспечения того, чтобы программные системы отвечали установленным требованиям. Эти методики накладывают различные требования в процессе разработки программного обеспечения с целью сделать этот процесс более эффективным и предсказуемым. Методики SDLC разделены на две группы (традиционные и легкие). Для разработки приложения в данной магистерской диссертации необходимо провести описание проблемы и следует провести исследования, какие цели должны быть достигнуты.

Создание приложения делится на 6 стадий (фаз), которые охватывают все процессы, необходимые для разработки компьютерной системы. Это позволяет принимать больший контроль над процессом на каждом этапе. Это также позволит сделать более точные прогнозы графиков проекта, таким образом предотвращая задержки проекта и перерасхода средств. Шесть этапов развития включают в себя: технико-экономическое обоснование; системный анализ; системный дизайн; реализация; контроль; обзор и техническое обслуживание. Каждая стадия четко определяет набор вспомогательных задач и процессов обеспечения того, чтобы все аспекты создания приложения были освещены. Поэтому каждый этап должен быть

завершен полностью, прежде чем перейти на следующий этап. Т.е. никакие две ступени не могут работать в асинхронном режиме.

Преимуществом использования этой модели является то, что пользователь участвует в создании и этапах реализации. На этапе исследования систем пользователь участвует в определении потребностей пользователей и потребностей системы, и далее на этапе осуществления для процедуры пользовательского тестирования и ввода данных. Участие пользователей имеет решающее значение для успеха системы.

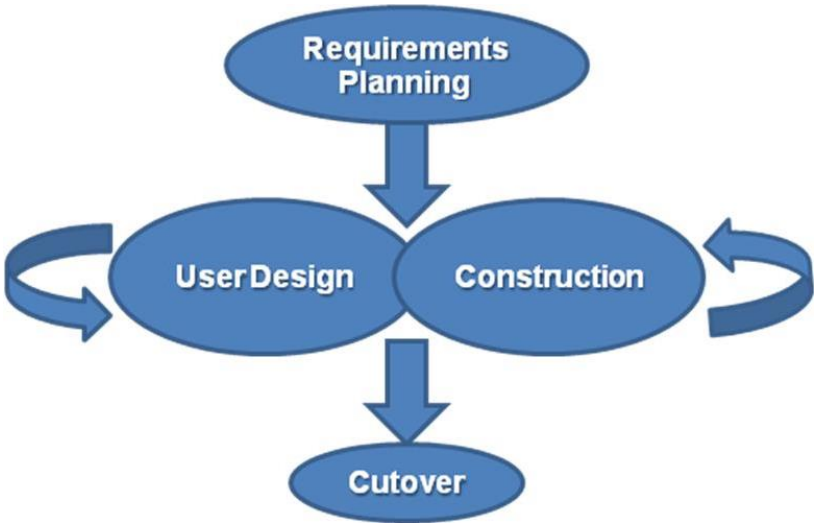


Рис. 3.1. Жизненный цикл информационной системы

Недостатком этой модели является отсутствие учета изменения в потребностях пользователей. Деление на 6 различных фаз делают модель негибкой при включении изменений потребностей пользователей в процессе разработки. Это может означать, что к концу развития система больше не имеет никакой реальной пользы для пользователя.

Адаптация этой модели, которая включает в себя возможность вернуться к предыдущему этапу для рассмотрения работы, может преодолеть данную проблему. Это решение известно как итерационная модель разработки ПО с циклами обратной связи между этапами. Эта модель

позволяет разработчику вновь вернуться к стадии проектирования стадии или набору требований, и включать любые изменения в требованиях пользователей, которые могут возникнуть в процессе разработки. Это, однако, может сделать более трудным график проекта и не будет ясного конца каждой стадии.

3.1.2 Разработка модели поисковой системы

В этой диссертации использована MVC модель. Модель-представление-контроллер (MVC) – архитектурный шаблон, который разделяет приложение на три основных логических компонента: модель, представление и контроллер. Каждый из этих компонентов строится для обработки конкретных аспектов приложения. MVC является наиболее часто используемой стандартной формой веб разработки для создания масштабируемых и расширяемых проектов.

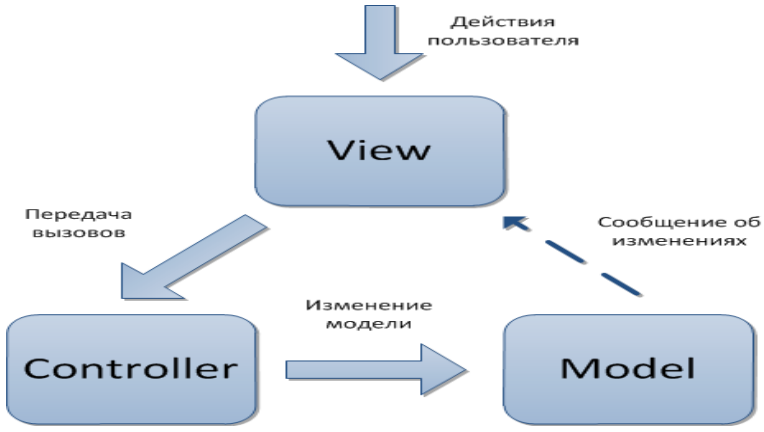


Рис. 3.2. Model-view-controller (MVC, «Модель-представление-поведение», «Модель-представление-контроллер»)

Модель (Model) предоставляет данные и реагирует на команды контроллера, изменяя свое состояние. Представление (View) отвечает за отображение данных модели пользователю, реагируя на изменения модели.

Контроллер (Controller) интерпретирует действия пользователя, оповещая модель о необходимости изменений

View Component (компонент представлений) используется для логики пользовательского интерфейса приложения. Например, компоненты представлений клиента будут включать в себя все компоненты пользовательского интерфейса: текстовых полей, раскрывающихся списков и т.д. Шаблоны хранятся в Папке resources/views. Они могут быть организованы в подпапки внутри этой директории.

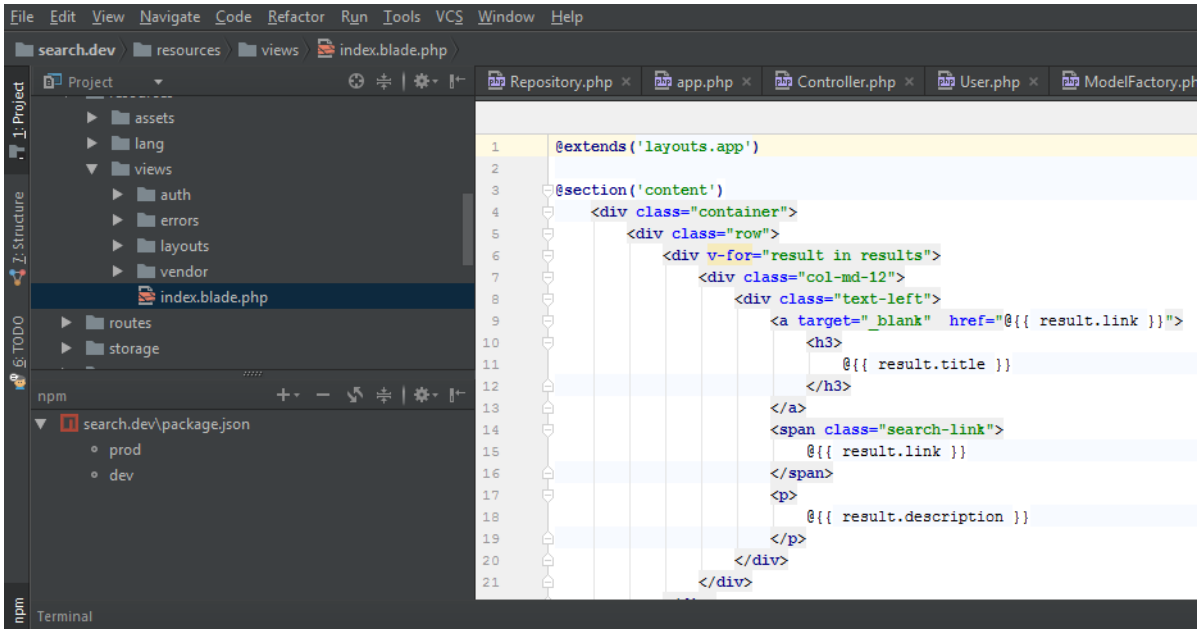


Рис. 3.3. Пользовательский интерфейс системы

Контроллеры действуют, как интерфейс между моделью и представлением компонентов для обработки всех логики и входящие запросы, манипулировать данными с использованием модели компонента и взаимодействовать с видом для окончательного вывода.

Листинг:

```
<?php
namespace App\Http\Controllers;
use Illuminate\Foundation\Bus\DispatchesJobs;
use Illuminate\Routing\Controller as BaseController;
use Illuminate\Foundation\Validation\ValidatesRequests;
use Illuminate\Foundation\Auth\Access\AuthorizesRequests;
```

```
class Controller extends BaseController
{
   use AuthorizesRequests, DispatchesJobs, ValidatesRequests;
}
```

3.1.3 Методология Эволюционное развитие проекта

Методология Эволюционное развитие является еще одним поэтапным подходом. Создание системы разделено на несколько итераций, проведенных одна за другой. Каждая итерация содержит этапы: требования, анализ, проектирование, внедрение. Хотя первая итерация будет включать в себя лишь небольшую часть требований, такой подход позволяет извлечь уроки из каждой итерации и улучшить в следующей итерации, таким образом, совершенствуя систему.

Эволюционное развитие особенно хорошо для проектов, где требования пользователей трудно заранее выяснить или могут меняться на протяжении всего процесса разработки. Этап «требования» в каждой итерации позволяет учесть изменение требований, которые должны быть рассмотрены и реализованы. В этом смысле он похож на прототипирование, и позволяет пользователю следить за развитием на всех этапах разработки.

К недостаткам такого подхода: не в состоянии удовлетворить полные потребности пользователей на ранней стадии, и трудности в управлении каждой итерацией.

Эта методика может быть также использован в сочетании с протоколом прототипирования, позволяя пользователю принимать участие на всех этапах разработки [6].

3.1.4 Прототипирование

Есть два типа методологий прототипирования: прототипирование и эволюционное прототипирование. Coad и Yourdan в работе [15] предлагают

"прототипирование следует использовать для всех объектно ориентированных проектов", но прототипирование в настоящее время используется в более широко. Его использование особенно растет при веб-разработке.

С прототипом не существует четкого этапа определения потребностей пользователей. Вместо того, чтобы прототип производился для пользователя, чтобы оставить комментарий, а затем либо усиливается, чтобы сформировать окончательную систему (эволюционное прототипирование) или отбрасывается после того, как были определены требования пользователей. Важно также отметить, что это может занять несколько прототипов, пока пользователь и разработчик согласовать набор требований, таким образом, возникает риск того, что соглашение не может быть достигнуто. Время разработки также может быть потрачено впустую из-за реализации вещей, которые пользователь позже решит не подходящими.

Преимущества этой методики: помимо вовлечения пользователя, она также дает возможность точно установить, что технологически осуществимо, экспериментируя с прототипом и обсудить это с разработчиком. Прототипы подходят для систем с небольшим числом пользователей и короткой продолжительности проекта, что не подходит для данной магистерской дисссертации. Кроме того, тестирование системы, разработанной с помощью прототипирования, делается все более трудным при отсутствии описания требований.

3.1.5 Структурированные системы анализа и проектирования (Ssadm)

Каскадная модель (англ. waterfall model, иногда переводят как модель «Водопад») - модель процесса разработки программного обеспечения, в которой процесс разработки выглядит как поток, последовательно

проходящий фазы анализа требований, проектирования, реализации, тестирования, интеграции и поддержки. Ssadm является хорошо структурированной структурой, которая охватывает все аспекты каскадной модели, за исключением внедрения и сопровождения. Она тоже разбивается на ряд этапов, каждый с четко определенным набором правил и руководящих принципов, включая Deliverables, необходимые на каждом этапе.

Ssadm фокусируется на анализе и проектировании системы и акцентирует внимание на потребностях пользователей, логической спецификации системы и вариантах технической системы. Эта методика обеспечивает тщательное изучение конструкции и лучше всего подходит для крупномасштабных проектов в области развития системы, требующих объемной документации, поэтому не подходит для целей данной диссертации.

3.2 Выбранная методология

С помощью анализа исследуемых методик стало очевидным, что целесообразно использовать модель прототипирования, и модель «водопад» (каскадную) с итеративной обратной связью, лучше всего подходящую для развития. В данной диссертации модель прототипирования будет использоваться в процессе привлечения пользователей и физического развития CMS, модель «водопад» четко определяет этап для установления требований пользователей, и это будет иметь решающее значение для успеха данного проекта.

По этой причине модель «водопад» с итеративной обратной связью будет использоваться для структурирования развития CMS. Итерационная обратная связь позволит на этапах создания программного продукта внести необходимые изменения с учетом потребностей пользователей.

ГЛАВА 4 ДИЗАЙН

4.1 Выбор дизайна

Дизайн является следующим шагом в модели «водопад» [14] и может иметь место только после того, как были собраны точно требования приложения. Итогом будет описание того, каким образом система будет включать в себя общие критические факторы успеха [2] и Веб-юзабилити сайта на основе метода «контрольный список». Применяемая конструкция будет проста в использовании и обслуживании для пользователя.

4.2 Данные

Данные, которые должны быть сохранены в этом приложении, будут храниться в базе данных MySQL, поскольку она находится в свободном доступе, и совместима с веб-серверами и выбранным языком - ASP. Для того, чтобы определить структуру базы данных, крайне важно, чтобы сначала выделить хранилища данных, необходимые для выполнения общих требований. Это потребовало создания таблицы схем, чтобы определить информацию для сохранения. Диаграмма Entity Relational затем может быть использована для представления общей структуры базы данных.

4.2.1 ER-модель данных

Диаграмма помогает проиллюстрировать концептуальное проектирование базы данных и показывает соотношения между элементами. Каждый объект в базе данных должны также иметь уникальный идентификатор для идентификации конкретных экземпляров каждого объекта. В примере таблицы Игроки это может быть PlayerID свойство, которое является уникальным для каждого игрока. Это поможет при

использовании SQL для запроса к базе данных. Диаграмма ER также помогает моделировать отношения между каждым объектом. Существуют три типа отношений, которые могут быть смоделированы: один-к-одному (1: 1); один ко многим (1: M); или многие-ко-многим (N: M). 1: 1 отношения, где один экземпляр объекта относится к одному экземпляру другого субъекта. A 1: M отношения означает экземпляр одной сущности может относиться к нескольким экземплярам другого субъекта. N: M отношения означают несколько экземпляров одного объекта может относиться к нескольким экземплярам другого субъекта.

4.2.2 Схема базы данных

Функциональность и содержание разрабатываемого приложения будет контролироваться базой данных, так что необходимо обеспечить, чтобы таблицы были созданы правильно. Схема базы данных будет выглядеть следующим образом (рис.4.1).

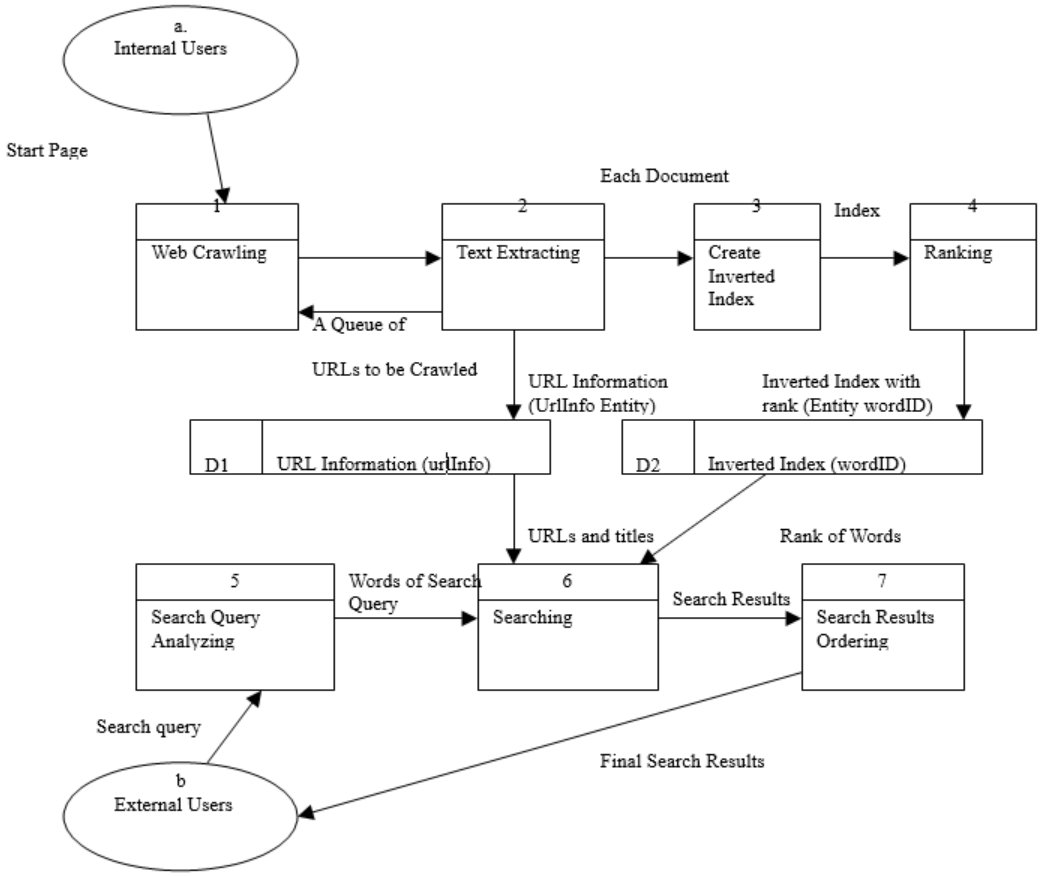


Рис. 4.1. Блок-схема логических данных

Обзор модели логического потока данных рисуется, как описано на (рис. 4.2).

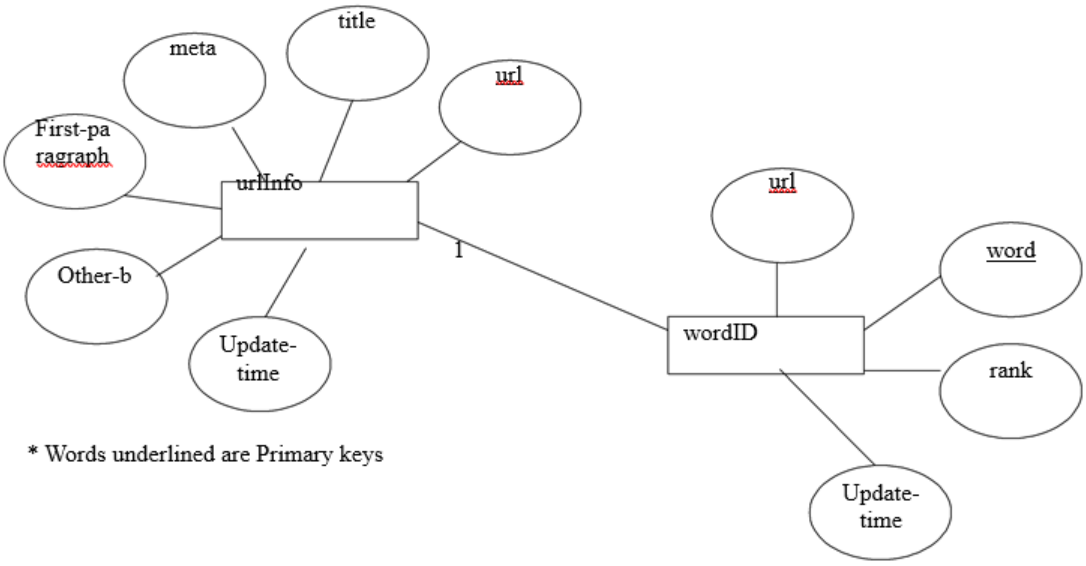


Рис. 4.2. Логическая модель данных

Модель показывает все функции / процессы, необходимые для предлагаемой системы. В этом разделе обеспечивается только обзор системы. Детали функций будут объяснены в физической конструкции. Есть 2 хранилища данных и 7 процессов в схеме логического потока данных.

4.2.3 Ограничения целостности

Таблицы в предлагаемой структуре базы данных связаны друг с другом посредством использования первичных ключей и внешних ключей в схеме базы данных. Каждая таблица в этой базе данных включает в себя первичный ключ, чтобы однозначно идентифицировать каждую запись в базе данных. Это необходимо, поскольку записи могут иметь одинаковое значение для полей, таких как "Имя". Первичный ключ в каждой таблице в этой структуре базы данных представляет собой числовой идентификатор поля типа AutoNumber, чтобы обеспечить запись и не имеет такой же идентификатор. Для того чтобы сформировать связь между двумя таблицами, при объединении таблиц должны также содержать первичный ключ таблицы, он формирует отношения. Этот ключ обычно применяют в качестве внешнего ключа.

4.3 Презентация

4.3.1 Веб-интерфейс

Пять особенностей дизайна интерфейсов, определенных Molly E Holzchlag, [5] подробно рассматривались в разделе исследований, при разработке веб-интерфейса для предлагаемого приложения. Конструкция также будет соответствовать юзабилити контрольного списка Томасона [2]. Молли Е Holzchlag [20] описывает 4 вида макета веб-страницы: с помощью стандартного HTML; таблицы на основе макета; макет на основе кадров; и

каскадных таблиц стилей (CSS). HTML чаще всего используется для структурирования текста и языка, в то время как таблица макетов на основе обеспечивают более доступный и гибкий вариант форматирования, чем фреймовых макетов.

CSS обычно используются для элементов дизайна, но могут возникнуть затруднения с браузером сочетаемости, поэтому, как правило, хорошо для позиционирования и стилей шрифтов. Другим фактором дизайна, требующим рассмотрения, является размеры страницы. Экраны компьютера, как правило, меньше, чем большинство открытых книг или журналов. Частая ошибка в веб-дизайне распространяется на ширину страницы графики за пределами области, которую большинство зрителей могут увидеть на своих экранах.

Согласно Руководству Web Style: 2-е издание [19] "безопасная область графики для веб-страниц определяется двумя факторами: минимальный размер экрана в общем пользовании и ширине бумаги, используемой для печати веб-страниц. Наиболее распространенным сегодня дисплеем является экран 800 x 600 пикселей.

Рис.4.3. Интерфейс поисковой системы

Веб-интерфейс будет построен по этим размерам с использованием таблиц для разметки, HTML для языка структурирования и CSS для управления позиционирование и стиль текста активов на каждой странице.

4.3.2 Цвета

Использование цветов для веб-сайта является очень важным. Хорошо известно, что высокий контраст очень важен для веб-юзабилити, таким образом, темные цвета должны быть использованы против светлых цветов, и наоборот. В соответствии с выбранной рамкой юзабилити, цвета должны в первую очередь отражать двигатель поиска, и читатель видит то, что он или она ищет и делает поиск проще и эффективнее.

4.3.3 Графика

Графика помогает пользователю понять, где ссылка или кнопка, но важно, чтобы использование графического изображения являлось для пользователя совершенно очевидным, и последовательно использовалось в приложении. Если пользователю нужно навести курсор на графический объект, он должен это увидеть. Автор, веб-дизайнер и юзабилити-аналитик Винсент Фландрии [21] описывает навигационные системы, где непомерно трудно для пользователей различать направления навигационных гиперссылок как Mystery Meat Navigation (MMN). Поэтому в диссертации принято решение использовать графику, которая поддерживается альтернативным текстом.

ГЛАВА 5 РЕАЛИЗАЦИЯ ПРИЛОЖЕНИЯ

5.1 Вводная часть

Этапом реализации проекта является разработка проектов. Через серию снимков экрана, фрагментов кода и описания в этом разделе будет показано, как были выполнены общие требования, и в случае необходимости различия между предлагаемыми проектами и фактической реализации.

Для реализации поисковика принято решение использовать следующее программное обеспечение: Composer (пакетный менеджер уровня приложений для языка программирования PHP, который предоставляет средства по управлению зависимостями в PHP-приложении), PHP-фреймворк laravel 5.4, jetbrain phpstorm 2016.3, в xampp-Win32 c-7.0.13-0-VC14-установщик, Node.js, Gulp,npm, vue.js 2.0. а также импортировать некоторые файлы из GitLab.

5.2 Установка и настройка

1. Предварительно установить Xampp, с Официальной страницы https://www.apachefriends.org/index.html.
2. установить Composer с Официальной страницы https://getcomposer.org/download/. После его установки, редактировать **httpd-vhosts.conf** , расположен в **C:\xampp\apache\conf\extra\httpd-vhosts.conf** и добавить следующие строки в конце файла:

```
# VirtualHost for LARAVEL.DEV
<VirtualHost laravel.dev:80>
  DocumentRoot "C:\xampp\htdocs\laravel\public"
  ServerAdmin laravel.dev
 <Directory "C:\xampp\htdocs\laravel">
      Options Indexes FollowSymLinks
```

AllowOverride All

Require all granted </Directory>

</VirtualHost>

После этого apache создаст laravel.dev соединения, требуется настроить хосты, и перенаправить файл laravel.dev на localhost, расположенный в C:\Windows\System32\drivers\etc

Редактировать файл hosts и сохранить:

```
# localhost name resolution is handled within DNS itself.
# 127.0.0.1 localhost
#:: 1 localhost
127.0.0.1  laravel.dev
```

3. Далее установить и настроить Laravel Framework. Во-первых, перейдите к htdocs папки, чтобы установить его и запустить следующую команду:

Composer create-project laravel/laravel laravel "5.1.*"

Начнется установка. Когда она закончится, будет создавать следующие схемы каталога:

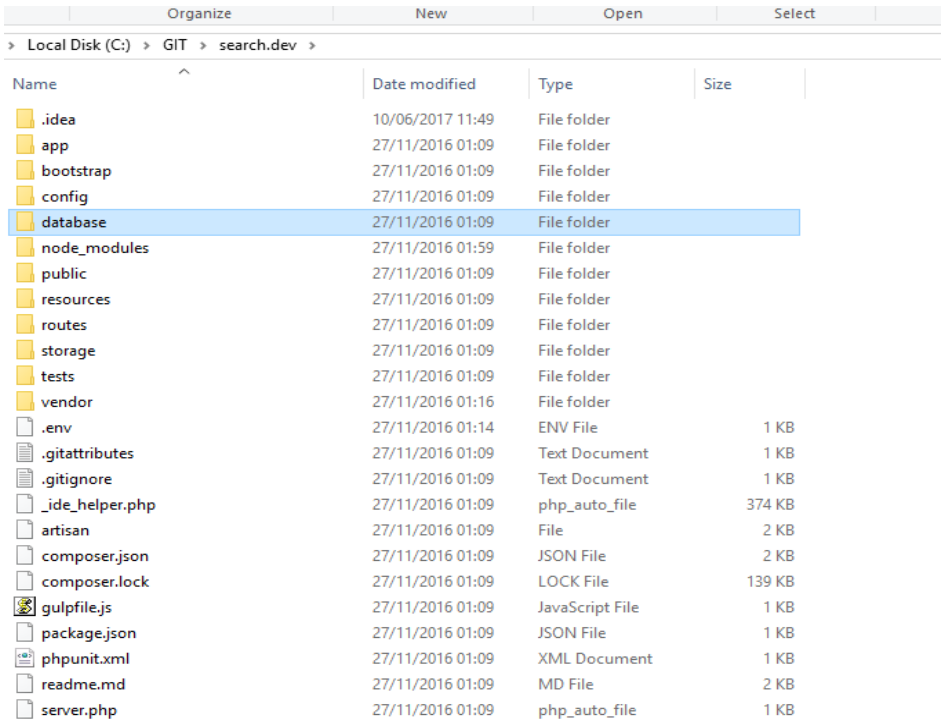


Рис. 5.1. схемы каталога

Наконец запустите apache и MySql из Панель управления XAMPP:

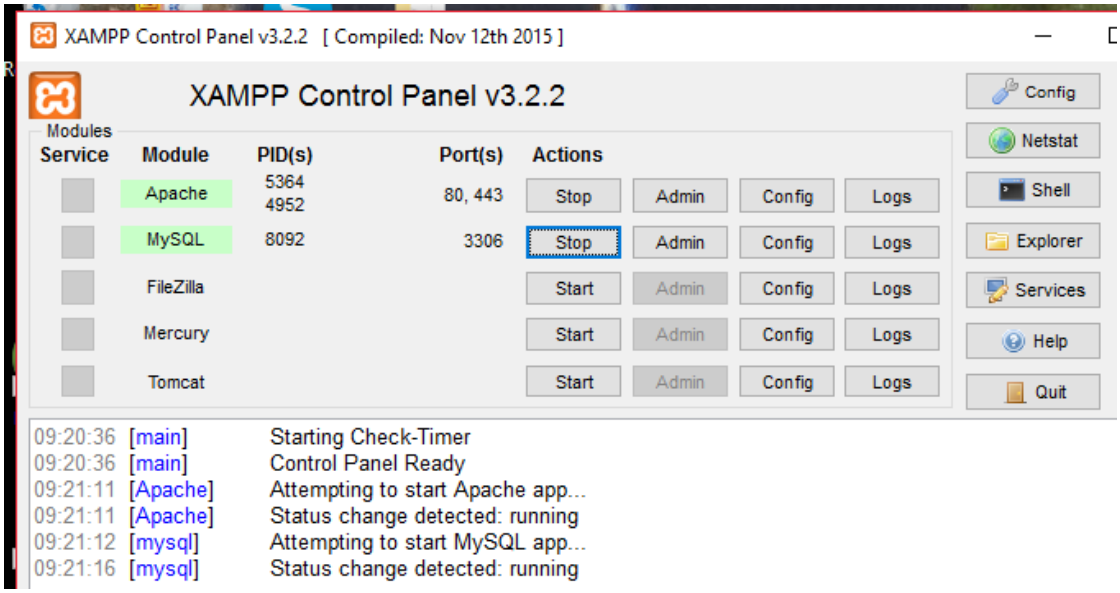


Рис. 5.2. Панель управления Xampp

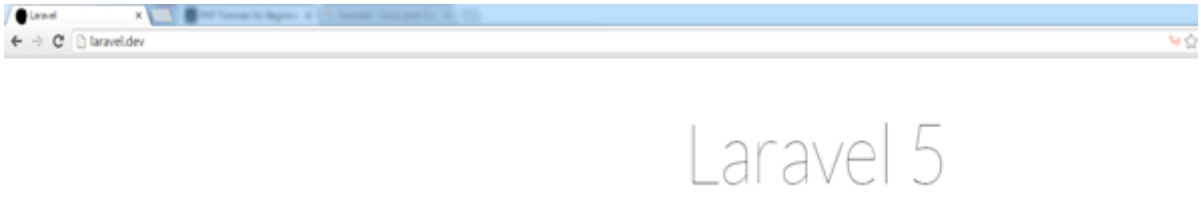


Рис. 5.2. Панель управления Xampp

Готово. Перейдите к **laravel.dev** и Laravel, он установлен.

5.3 Необходимые условия

Инициализация Composer

Composer инициализируется и настраивается в PhpStorm. После открытия Laravel проектом, выбран корневой узел в окне инструментов проекта и используется Composer | ***Init Composer...*** контекстное меню. PhpStorm можно скачать *composer.phar* при необходимости.

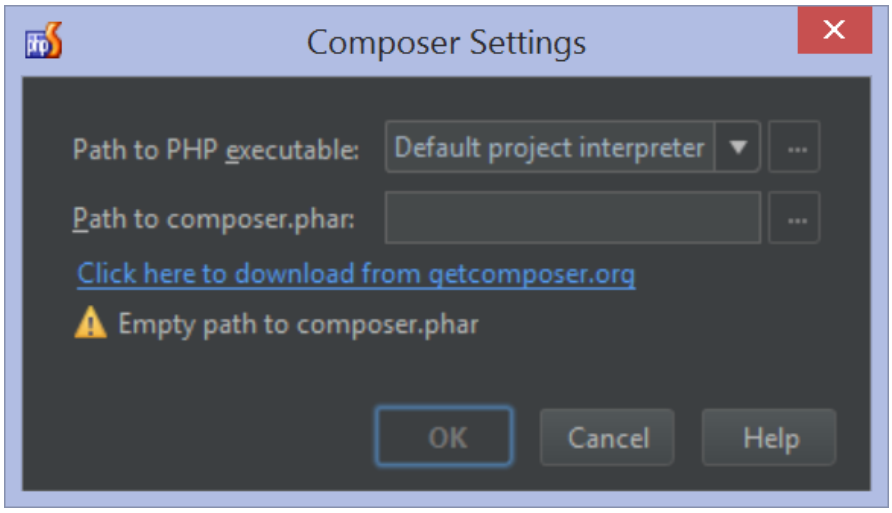


Рис. 5.4. Настройка композитора.

Установка хелпер Laravel IDE

Чтобы установить модуль поддержки IDE laravel, был использован ***Composer | Add dependency...*** (***Composer | Добавление зависимостей)*** контекстное меню и искали ***barryvdh/laravel-ide***-helper. Нажмите **Установка** скачать пакет и добавить его к проекту.

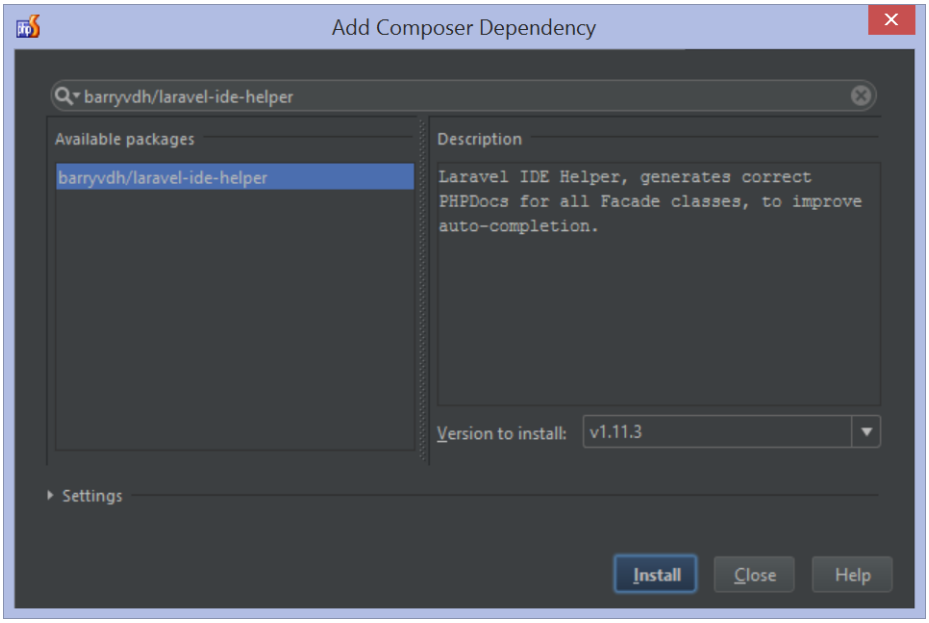


Рис. 5.5. Добавление зависимостей

После установки добавьте хелпер Laravel IDE как *ServiceProvider* в приложении. В *app/config/app.php* файл, был добавлен 'Barryvdh\ Laravel Ide Helper\ Ide Helpe Service Provider' под *providers* элемент, как показано:

```
<?php
return array(
    'providers' => array(
        'Barryvdh\LaravelIdeHelper\IdeHelperServiceProvider', // Laravel IDE helper
    ),
);
```

Создание PHPDoc вспомогательный файл

Самый простой способ сделать это - включить поддержку инструмента командной строки для *artisan*. Из настроек было добавлено новое средство командной строки под ***Tools | Command Line Tool Support (инструменты | Поддержка командной строки инструмента)***. Далее, укажите путь к интерфейсу командной строки *artisan*.

После сохранения, используйте *artisan* из среды IDE. ***Инструменты | Выполните команду...*** меню (***Ctrl + Shift + X*** или ***CMD + Shift + X*** на Mac OS X) обеспечивает завершение для всех доступных команд ремесленника. Запуск artisan ide-helper:generate команда для создания требуемой информации PHPDoc.

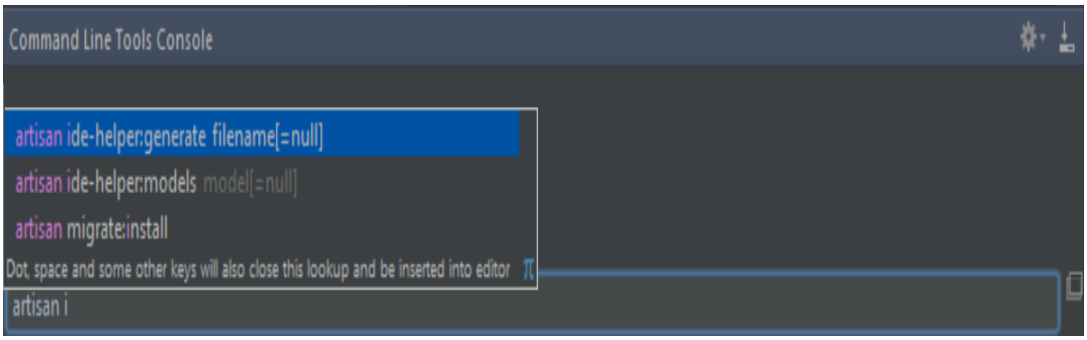


Рис. 5.6. Выполнение команды в artisan ide-helper

Хелпер Laravel IDE (Laravel IDE Helper) должен выполняться после изменения или добавления услуг, контроллеры, модели и представления.

Установите и включите плагин Laravel

Параметры (настройки) | Плагины, нажмите **Просмотр хранилищ...** кнопку и поиск для *Laravel*. Далее можно использовать кнопку «**установить плагин**», чтобы установить плагин.

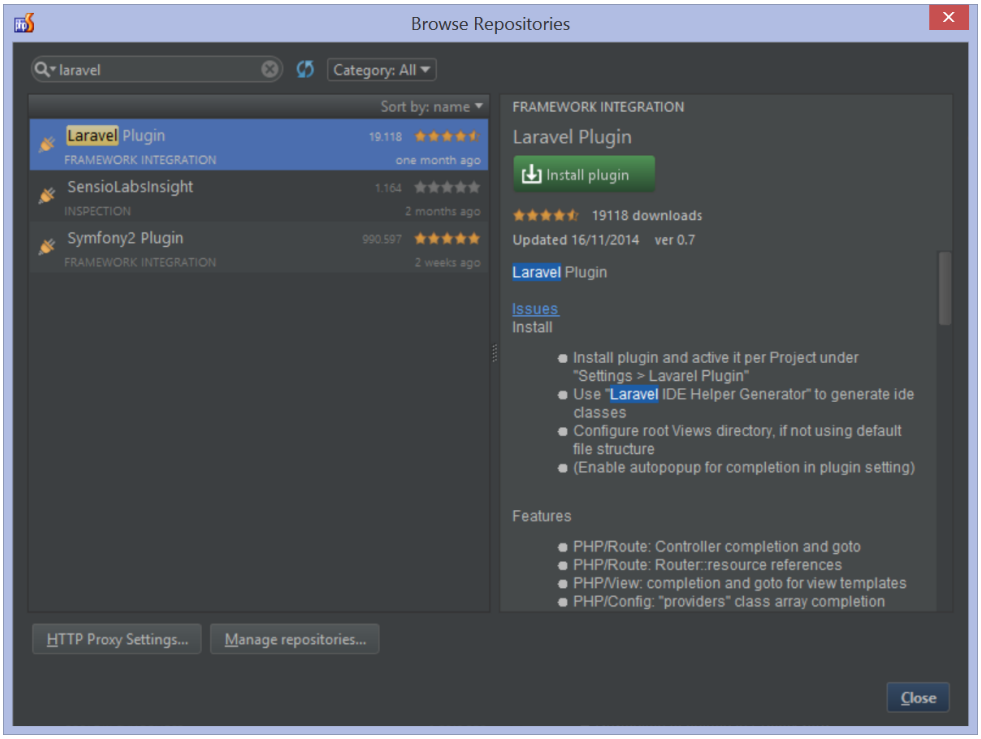


Рис. 5.7. установить плаги

Перезапуск IDE для завершения установки плагинов. Далее, включили *Laravel плагин* в ***Параметры (настройки)*** | ***Другие параметры*** | ***Laravel плагин*** | ***Включить плагин для этого проекта***. Перезапустить среду IDE снова, чтобы загрузить дополнительные функции плагина.

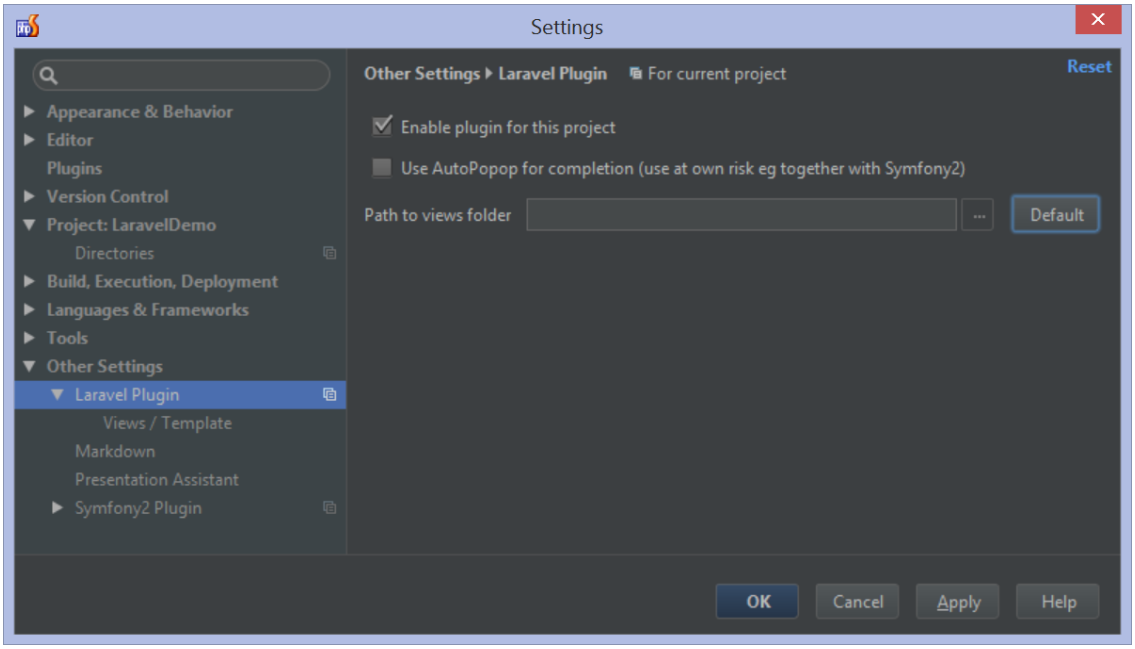


Рис. 5.8. Включить плагин для этого проекта

В случае возникновения проблем с завершением и навигацией поддержки плагинов, выберите *Файл | Недействительный кэш / перезапустить* для индексации проекта. Могут также потребоваться запуски artisan clear-compiled и artisan ide-helper:generate.

Интеграция XAMPP с IDE

Необходимо указать IDE, где эти компоненты хранятся и как они настроены.

Интеграция исполняемого PHP

Зарегистрировать исполняемый файл PHP с XAMPP в PhpStorm, это можно сделать с помощью **Файл | Параметры** меню ***(Ctrl + Shift + S)***, навигационные для **Параметры проекта | PHP**.

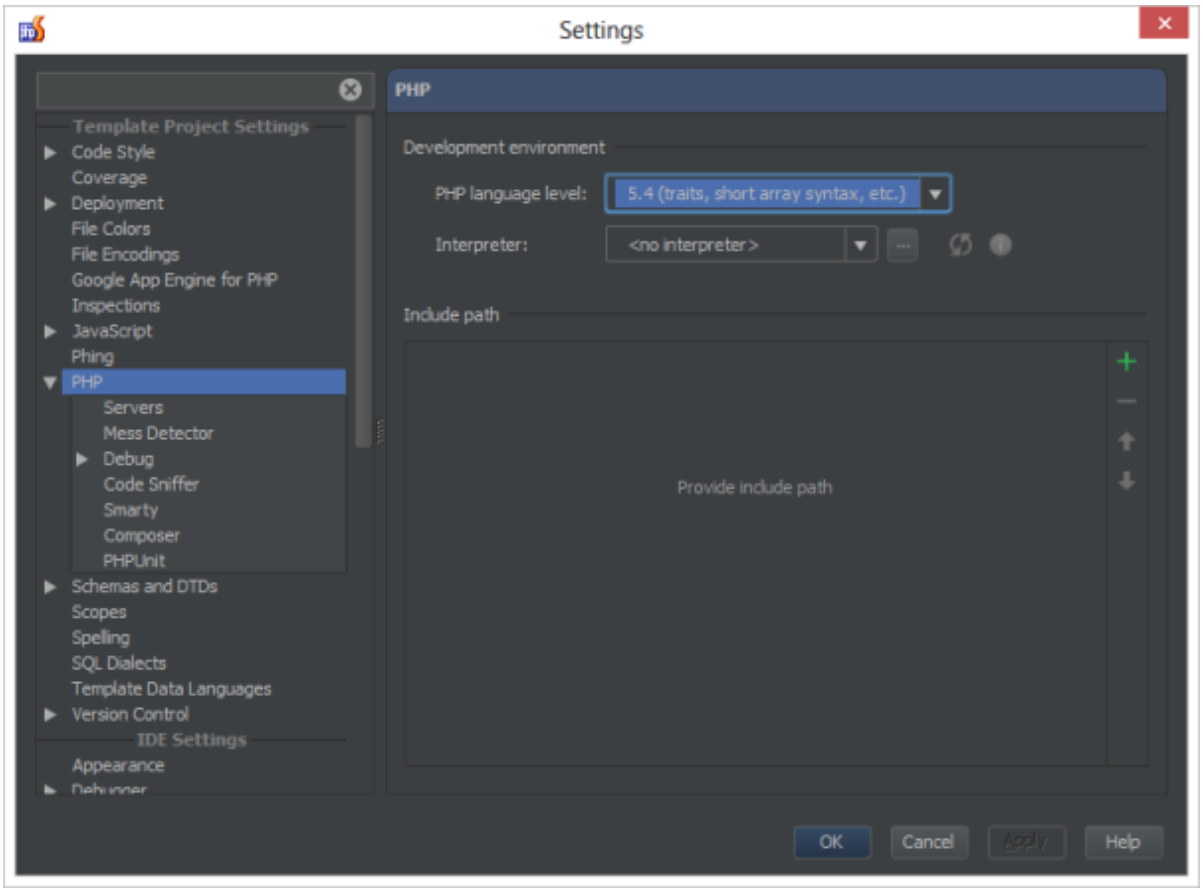


Рис. 5.9. Интеграция исполняемого PHP

С IntelliJ IDEA и PhpStorm можно иметь несколько отдельных PHP переводчиков, зарегистрированных в среде IDE, в зависимости от версии PHP и/или конфигурации PHP, необходимой для проекта.

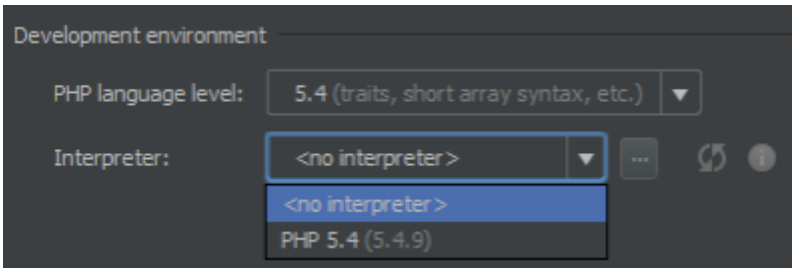


Рис. 5.10. среда разработки

Можно добавить один: установили с XAMPP, нажав кнопку "Обзор" (**...**).

1. В левой панели нажмите кнопку **Добавить.**
2. В текстовое поле **имя**, введите имя для идентификации текущей установки, например *PHP с XAMPP*.
3. В текстовом поле **PHP home** укажите папку, где исполняемый файл PHP *php.exe* хранится. Поскольку мы установили XAMPP для *c:\xampp* домашний каталог PHP будет *C:\xampp\php*. Мы можем ввести этот путь вручную или использовать кнопку **Просмотр**, чтобы найти путь в нашей системе.
4. IDE проверяет, содержит ли указанная папка исполняемый файл PHP, определяет версию PHP и отображает его в поле **PHP информация** только для чтения.

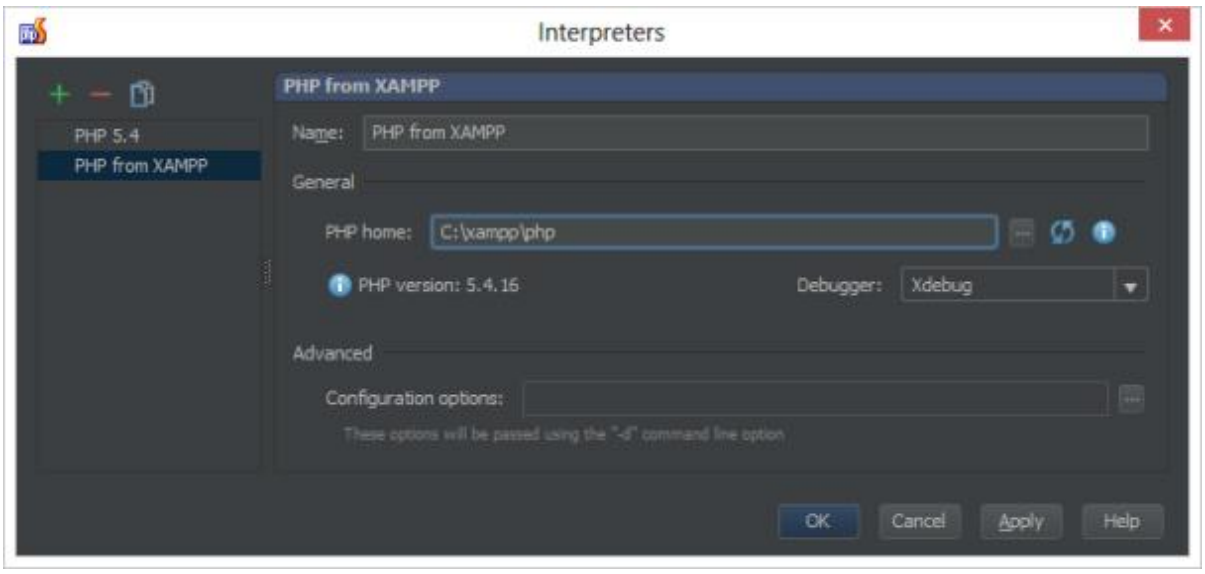


Рис. 5.11. исполняемый файл PHP

Интеграция сервера Apache

Взаимодействие между IntelliJ IDEA или PhpStorm с Web, FTP и других серверов поддерживается через плагин Удаленного доступа хостов, который по умолчанию включен для IntelliJ IDEA и PhpStorm. IDE обращается к серверу с помощью параметров подключения, указанных в конфигурации доступа зарегистрированного сервера. Эти настройки создаются и управляются в **Параметры проекта | Развертывания** страница **Параметры** диалоговом.

Выбрать **Файл | Параметры** в главном меню снова, чтобы открыть **Параметры**. Нажмите **развертывание** под узел **Параметры проекта**, чтобы перейти к **развертыванию** страницы.

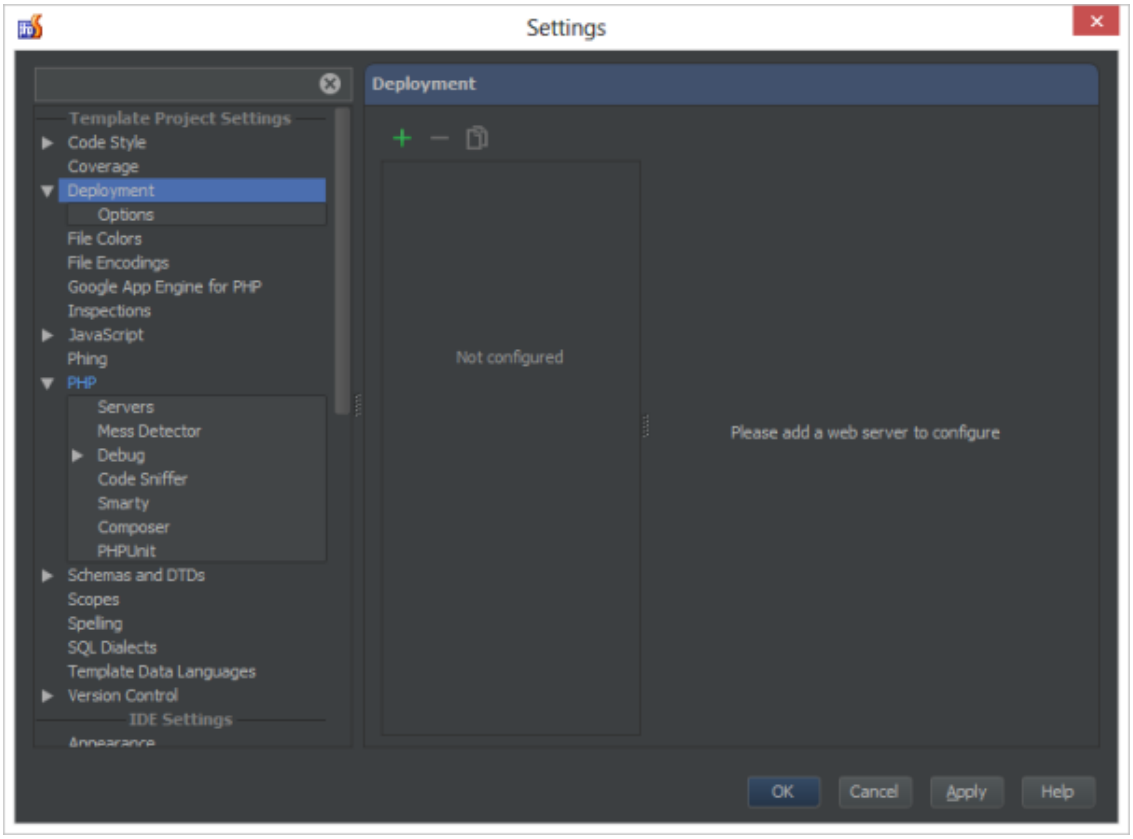


Рис. 5.12. развертыванию страницы (установка)

Из панели инструментов, нажмите кнопку **Добавить**, чтобы добавить новый сервер. От **Добавить сервер** диалоговом , можно указать имя для сервера XAMPP и выбрать тип развертывания. В XAMPP Apache сервера, выберите тип *местные или подключенные папки*.

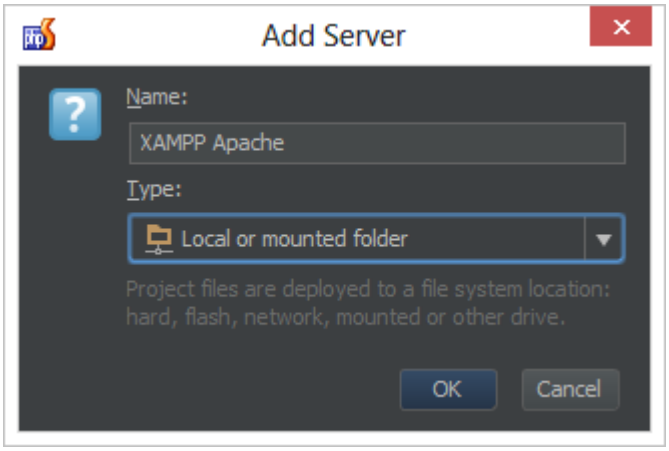


Рис. 5.13. Добавить сервер

После нажатия кнопки **ОК,** можно указать папку для Apache web корня (*C:\xampp\htdocs*) и URL-адрес веб-сервера, *http://localhost*. Обратите внимание, что эти параметры могут отличаться в зависимости от того, как настроили сервер XAMPP Apache через *C:\xampp\apache\conf\httpd.conf* файла конфигурации.

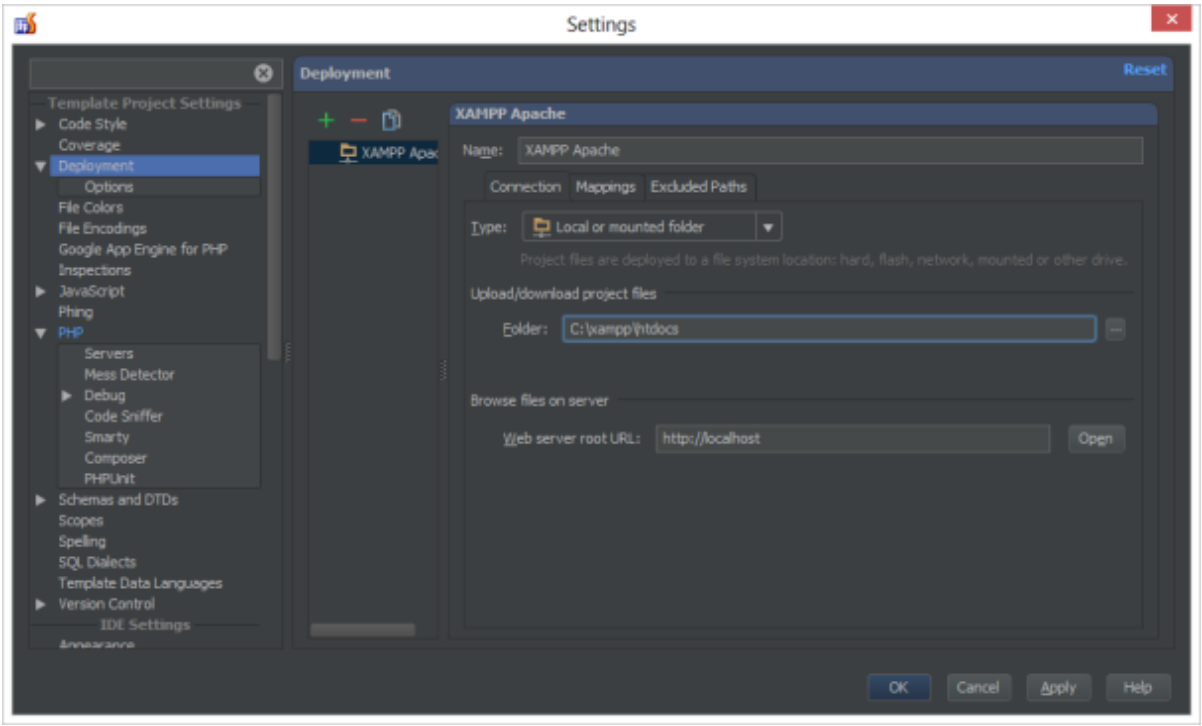


Рис. 5.14. папку для Apache web корня

С использованием кнопки **Открытые**, можно проверить параметры ОК. После этого, пользователи должны видеть на главной странице XAMPP. Blade шаблонов

Реализация Blade

Blade — простой, но мощный шаблонизатор, поставляемый с Laravel. В отличие от других популярных шаблонизаторов для PHP, Blade не ограничивает в использовании чистого PHP-кода в требуемых представлениях. Все представления Blade скомпилированы в чистый PHP-код и кешированы, пока в них нет изменений, а значит, Blade практически не нагружает ваше приложение.

Для этого проекта использован следующий код

```
<html lang="en">
<head>
   <title>Search Engine</title>
   <meta name="title" content="Search Engine">
   <meta charset="utf-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1">
   <meta name="csrf-token" content="{{ csrf_token() }}">
   <link href="/css/app.css" rel="stylesheet" />
</head>
<body>
   <nav class="navbar navbar-default navbar-static-top">
      <div class="container">
         <div class="navbar-header">
            <button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-
target="#app-navbar-collapse">
               <span class="sr-only">Toggle Navigation</span>
               <span class="icon-bar"></span>
               <span class="icon-bar"></span>
               <span class="icon-bar"></span>
            </button>
            <a class="navbar-brand" href="{{ url('/') }}">
               Search Engine
            </a>
         </div>
         <div class="collapse navbar-collapse" id="app-navbar-collapse">
            <ul class="nav navbar-nav">
                
                     </ul>
```

Реализация Guzzle, PHP HTTP клиента

Guzzle является PHP-HTTP-клиентом, который позволяет легко отправлять HTTP-запросы и тривиальные для интеграции с веб-службами. В диссертации использованы следующие коды для отправки запроса

```
<?php
   namespace App\Repositories\Se;
use GuzzleHttp\Client;
use GuzzleHttp\Cookie\FileCookieJar;
use GuzzleHttp\Promise;
  class Repository
 {
   protected $client;
   protected $cookie;
   protected $cookie_name = 'se.txt';
   protected $cookie_path;
```

```
    protected $user_agent = 'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/54.0.2840.87 Safari/537.36 OPR/41.0.2353.56';
    public function __construct()
    {
        $this->cookie_path = $this->getCookiePath();
        $this->cookie = $this->getCookie();
        $this->client = new Client([
            'cookies' => $this->cookie,
            'headers' => [
                'Accept-Encoding' => 'gzip, deflate, lzma, sdch',
                'Accept-Language' => 'ru-RU,ru;q=0.8,en-US;q=0.6,en;q=0.4',
                'DNT' => '1',
                'User-Agent' => $this->user_agent
            ]
        ]);
    }
    public function __destruct()
    {
        $this->saveCookie();
    }
    protected function getCookiePath(): string
    {
        return storage_path('app') . '/' . $this->cookie_name;
    }
    protected function getCookie(): FileCookieJar
    {
        return new FileCookieJar($this->cookie_path);
    }
    protected function saveCookie()
    {
        $this->cookie->save($this->cookie_path);
    }
    protected function getContentFromBing($query)
    {
        $pages = [0, 10]; $promises = [];
        foreach ($pages as $page) {
            $promises[] = $this->client->getAsync('http://www.bing.com/search', [
                'headers' => [
                    CURLOPT_SSL_VERIFYPEER => FALSE,
                    CURLOPT_SSL_VERIFYHOST => FALSE,
                    CURLOPT_FTP_SSL => CURLFTPSSL_TRY
                ],
                'query' => [
                    'first' => $page * 10,
                    'format' => 'rss',
                    'q' => $query
                ]
            ]);
        }
        return Promise\settle($promises)->wait();
    }
```

```
    public function getFromSearchEngine($query)
    {
        $responses = $this->getContentFromBing($query);
        $results = [];
        foreach ($responses as $response) {
            $page = simplexml_load_string((string)$response['value']->getBody());
            foreach ($page->channel->item as $item) {
                $results[] = [
                    'description' => str_replace('...', '', (string)$item->description),
                    'link' => (string)$item->link,
                    'title' => (string)$item->title
                ];
            }
        }
        return $results;
    }
}
```

Вывод к разделу

Для реализации поисковика принято решение использовать следующее программное обеспечение: Composer (пакетный менеджер уровня приложений для языка программирования PHP, который предоставляет средства по управлению зависимостями в PHP-приложении), PHP-фреймворк laravel 5.4, jetbrain phpstorm 2016.3, в xampp-Win32 c-7.0.13-0-VC14-установщик, Node.js, Gulp,npm, vue.js 2.0. а также импортировать некоторые файлы из GitLab.

ГЛАВА 6 ТЕСТИРОВАНИЕ СОЗДАННОГО ПО

Тестирование программного обеспечения является исследованием, проведенным для предоставления заинтересованным сторонам информации о качестве продукта или услуги в рамках теста. [1] Тестирование программного обеспечения может также дать объективное, независимое представление программного обеспечения, чтобы позволить бизнесу оценить и понять риски внедрения программного обеспечения. Методы испытаний включают в себя процесс выполнения программы или приложения с целью обнаружения ошибок в программном обеспечении (ошибки или другие дефекты), а также для проверки того, что программный продукт, пригодный для использования. Тестирование программного обеспечения включает в себя выполнение программного компонента или компонента системы для оценки одного или нескольких свойств, представляющих интерес. В общем, эти свойства указывают на степень, в которой компонент или тестируемая система:

- отвечает требованиям, которыми руководствовались его проектирование и разработку,
- правильно отвечает на все виды входов
- выполняет свои функции в течение приемлемого времени,
- достаточно удобны
- может быть установлена и запущена в ее предполагаемую среду
- достигает общего результата заинтересованные стороны желают.

6.1 Проверка Юзабилити

Оценка юзабилити основывается на общих принципах дизайна или конкретного перечня руководящих принципов и, как правило, осуществляется разработчиком интерфейса или разработчика. Недостаток такого подхода состоит в том, что оно не связано с пользователями, так что неясно, если проблема, выявленная разработчиком является проблемой для пользователя. По этой причине тестирование пользователей по-прежнему решающее значение для успешного завершения тестирования.

Проверка юзабилити дала в основном положительные результаты. Она показала, что расположение и конструкция удовлетворяют всем требованиям удобства и простоты использования в контрольном списке и что навигация в приложении реализована хорошо. Приложение также успешно выполнило все рекомендации, HCI (взаимодействия человека с компьютером) в контрольном списке. Использование цвета и содержания были также определены как хорошо. Осмотр, однако, подчеркнул слабость толерантности к ошибкам. Пользователям необходимо администрирование сайта, чтобы быть предупрежденными от рискованного или дорогостоящего действия, прежде чем они идут и нужен источник помощи, в случае возникновения ошибок.

6.2 Тестирование пользователем

Пользовательское тестирование направлено на выявление конкретных проблем и включает в себя наблюдения пользователей, выполняющих конкретные мероприятия с целью выявления того, что у них проблемы, как они используют приложение. Тестирование пользователей в этом проекте будет сосредоточено на тестировании интерфейса презентации потенциальным пользователем веб-сайта.

Вывод к разделу

Тестирование удобства использования показало, что пользователь получил результаты, предполагая, полученным результатам можно доверять и корректирующие меры должны быть приняты для устранения выявленных проблем. Кроме того, можно сделать вывод из пользовательского тестирования, что наиболее общие требования выполнены.

ЗАКЛЮЧЕНИЕ

Поисковые системы предлагают пользователям обширное и впечатляющее количество информации, скорость и удобство, которые десятилетие назад мало кто мог себе представить. Этот проект продемонстрировал универсальную поисковую систему, формирующую ядро приложений, которые облегчают процесс поиска информации в среде. Он использует алгоритм поиска стратегии, которая позволяет в будущем возможность модификации и усовершенствования. Этот подход использует все доступные вычислительные ресурсы эффективно по сравнению с большинство задач, выполняемых конечными серверами.

Одной из рекомендаций является поиск возможности регулярно обновлять алгоритм (должен быть проверен и периодически пересматриваться). Будущее направление этого исследования может быть продлено с помощью гибридных подходов.

Аналогичным образом, поисковые системы естественно будут продолжать развиваться на основе алгоритмов ранжирования и повышать релевантность результатов поиска - процесс, который органически приведет к решению наиболее проблемных аспектов поиска.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Langville, N.A & Meyer, D.C.(2012).*Google's pagerank and beyond: the science of search engine rankings.* Princeton, NJ, USA: Princeton University Press [Электронный ресурс]/ Режим доступа: http://geza.kzoo.edu-/~erdi/patent/langvillebook.pdf (дата обращения 05.01.2017г.)

2. Thomason, Larisa, (2004), Web Site Usability Checklist [Электронный ресурс]/ Режим доступа:http://www.netmechanic.com/news/vol7/design_no4.htm (дата обращения 07.01.2017г.)

3. Brink, Tom, Gergle, Darren, Wood, Scott, (2002), Usability Evaluation, in: Usability for the web: designing websites that work, pp. 405-441[Электронный ресурс]/ Режим доступа: http://www.informationr.net/ir/reviews/revs074-.html#book2 (дата обращения 14.01.2017г.)

4. Levene, M. (2010). *An introduction to search engines and web navigation*. 2nd edition, Harlow, England; New York: Addison-Wesley[Электронный ресурс]/ Режим доступа: http://proquest.tech.safaribooksonline.de/9780470526842-?uicode=CERN (дата обращения 04.02.2017г.)

5. Михнюк Д.В., Егошина А.А. Анализ современных тенденций использования коллаборативной фильтрации в веб-приложениях [Электронный ресурс]/ Режим доступа: http://seminar.at.ispras.ru/wp-content/uploads/2012/07/Gomzin-thesis.pdf (дата обращения 15.03.2017г.)

6. Интернет-маркетинг [Электронный ресурс]/ https://ru.wikipedia.org - Википедия – свободная энциклопедия// URL: https://ru.wikipedia.org-/wiki/Интернет-маркетинг (дата обращения 20.04.2017г)

7. Holzschlag, Molly E, (1998). The Art if Interface, in: *Web by Design: the complete guide*. pp. 77-83, San Francisco : Sybex [Электронный ресурс]/ Режим доступа: http://isbnbase.ru/Web-by-design--the-complete-guide--or--cMolly-E-Holzschlag/6/bajjgcg (дата обращения 19.02.2017г)

8. Nakano, Russell, (2002). *Web Content Management: a collaborative approach*. Mishawaka, IN, U.S.A: Addison- Wesley[Электронный ресурс]/ Режим доступа: http://isbnbase.ru/Web-content-management--a-collaborative-approach--or--cRussell-Nakano/6/bbajhcc (дата обращения 25.02.2017г)

9. Büttcher, S., Cormack,G.V & Clarke, C.L (2010). *Information retrieval: implementing and evaluating search engines.* Cambridge, Massachusetts

London, England; The MIT Press[Электронный ресурс]/ Режим доступа: https://mitpress.mit.edu/sites/default/files/titles/content/9780262026512_pre_0001.pdf(дата обращения 27.02.2017г)

10. Jones, Russel, A, (2000). *Behind the Scenes – How Active Server Pages Work, in: Mastering Active Server Pages* 3, pp. 4-32, Sybex[Электронный ресурс]/ Режим доступа: http://kek.ksu.ru/eos/ECOMMERCE/masteringasp/01-01.html (дата обращения 05.03.2017г)

11.Belkin, N. J., & Crof, W. B. (1992).*Information fltering and information retrieval: Two sides of the same coin? Communications of the ACM, 35*(12),29–38[Электронный ресурс]/ Режим доступа: http://maroo.cs.umass.edu/getpdf.-php?id=131 (дата обращения 05.03.2017г)

12. Brin, S. & Page, L. *The Anatomy of a Large-Scale Hypertextual Web Search Engine.* Computer Networks and ISDN Systems, 33:107–17, 1998[Электронный ресурс]/ Режим доступа: http://infolab.stanford.edu/~backrub/google.html (дата обращения 07.03.2017г)

13. Jepson, B, PostgreSQL vs. MySQL: building better databases, [Электронный ресурс]/ Режим доступа: www.webtechniques.com/archives/2001/09/jepson/ (дата обращения 13.03.2017г)

14. Avison, David, Shah, Hanifa, (1997). *Information System Development life cycle, in: The Information Systems Development Life Cycle: A first course in Information Systems*, pp.67-87, McGraw-Hill[Электронный ресурс]/ Режим доступа:https://www.google.ru/url?sa=t&rct=j&q=&esrc=s&source=web&cd=5&cad=rja&uact=8&ved=0ahUKEwjyypbniMrUAhVCYJoKHTrfCYQQFghGMAQ&url=http%3A%2F%2Fwww.springer.com%2Fcda%2Fcontent%2Fdocument%2Fcda_downloaddocument%2F9781461492535-c2.pdf%3FSGWID%3D0-0-45-1479416-p175478101&usg=AFQjCNEmgbSgN8VjSJYfdwJlER4_i8Y59A (дата обращения 23.03.2017г)

15. Avison, David, Fitzgerald, Guy, (2002). *Methodologies, in: information systems development, methodology, techniques and tools*, 3rd ed, pp. 347-433, McGraw-Hill[Электронный ресурс]/ Режим доступа: http://www.banrepcultural.org/sites/default/files/juan-manuel-munos-Metodologias_de_desarrollo_de_software_para_los_sistemas_de_informacion_de_apoyo_a_la_toma_de_decisiones_basados_en_datos.pdf (дата обращения 23.03.2017г)

16. Ricardo Baeza-Yates and Emilio Davis. Web page ranking using link attributes.In *The Thirteenth International World Wide Web Conference*, pages

328–29, New York, 2004. ACM Press. Poster[Электронный ресурс]/ Режим доступа: http://www2004.org/ (дата обращения 27.03.2017г)

17. Michael W. Berry and Murray Browne. *Understanding Search Engines: Mathematical Modeling and Text Retrieval*. SIAM, Philadelphia, 2nd edition, 2005[Электронный ресурс]/ Режим доступа: http://www.gbv.de/dms/-ilmenau/toc/269298568.PDF (дата обращения 28.03.2017г)

18. Arasu, A., Novak, J.,Tomkins, A. & Tomlin, J. *PageRank computation and the structure of the Web: Experiments and algorithms. In The Eleventh International WWW Conference*, New York, May 2002. ACM Press[Электронный ресурс]/ Режим доступа: http://dl.acm.org/citation.cfm?id=511446&picked=prox (дата обращения 27.03.2017г)

19. Gillies, J. & Cailliau,R. (2000). *How the Web Was Born: The Story of the World Wide Web*. Oxford University Press[Электронный ресурс]/ Режим доступа: http://fdslive.oup.com/www.oup.com/academic/pdf/-13/9780192862075.pdf prox (дата обращения 30.03.2017г)

20. Holzschlag, Molly E, (1998), *Layout Technology, in: Web by Design: the complete guide!* pp. 375-395, Sybex[Электронный ресурс]/ Режим доступа: http://isbnbase.ru/Web-by-design--the-complete-guide--or--cMolly-E-Holzschlag/6/bajjgcg (дата обращения 19.02.2017г)

21. Flanders, Vincent, Websites that suck [Электронный ресурс]/ Режим доступа: http://www.webpagesthatsuck.com/mysterymeatnavigation.html (дата обращения 22.04.2017г)

22. Web Wiz Guide, Web Wiz Rich text Editor, [Электронный ресурс]/ Режим доступа: http://www.webwizguide.info/asp/default.asp (дата обращения 23.04.2017г)

23. The ASP Resource Index [Электронный ресурс]/ Режим доступа: www.aspin.com (дата обращения 23.04.2017г)

24. Xiaoyan Zhang, Michael W. Berry, and Padma Raghavan. Level search schemes for information filtering and retrieval. *Information Processing and Management*, 37:313–34, 2001 [Электронный ресурс]/ Режим доступа: http://www.cse.psu.edu/~pxr3/Papers/ipm.pdf (дата обращения 01.05.2017г)

25. Coad, P. (1999). *Feature-Driven Development. Object International. Retrieved* April 8, 2001, from the World Wide Web [Электронный ресурс]/

Режим доступа: http://www.togethersoft.com/jmcu/chapter6.PDF(дата обращения 23.04.2017г)

26. Nunes, N. & Cunha, J. (2000). Wisdom: A software engineering method for small software development companies. *IEEE Software, September/October 2000*, 113-119 [Электронный ресурс]/ Режим доступа: http://toalango.com/msc/in-the-small.pdf (дата обращения 02.05.2017г)

27. Pollmann, T., & Baayen, R. H. (2001). Computing historical consciousness. A quantitative inquiry into the presence of the past in newspaper texts. *Computers and the Humanities,* 35, 237–253 [Электронный ресурс]/ Режим доступа: https://elibrary.ru/item.asp?id=794916 (дата обращения 05.05.2017г)

28. Herbert A. Simon. A Behavioral Model of Rational Choice. *The Quarterly Journal of Economics,* Vol. 69, No. 1. (Feb., 1955), pp. 99-118 [Электронный ресурс]/ Режим доступа: http://www.math.mcgill.ca/vetta/CS764.dir/bounded.pdf (дата обращения 06.05.2017г)

29. Stewart, N., Chater, N., & Brown, G.D.A. (2006). *Decision by sampling*. Cognitive Psychology, 53, 1–26[Электронный ресурс]/ Режим доступа:http://www.dectech.co.uk/publications/LinksNick/Reasoning-AndDecisionMaking/DbSCogPsychJournal.pdf (дата обращения 06.05.2017г)

30. Stewart, Neil. (2009) *Decision by sampling: the role of the decision environment in risky choice.* Quarterly Journal of Experimental Psychology, Vol.62 (No.6). pp. 1041-1062. ISSN 1747-0218 [Электронный ресурс]/ Режим доступа: http://wrap.warwick.ac.uk/706/ (дата обращения 06.05.2017г)

31. Newman M. E. (2005). Power laws, Pareto distributions and Zipf's law. *Contemporary Physics*, 46, 323–351[Электронный ресурс]/ Режим доступа:http://www2.math.uu.se/~david/web/SCDS/Newman05.pdf(дата обращения 07.05.2017г)

32. Kello C. T., Brown G. D. A., Ferrer- i- Cancho R., Holden J. G., Linkenkaer- Hansen K., Rhodes T. & Van Orden G. C. (2010). Scaling *laws in cognitive sciences. Trends in Cognitive Sciences*, 14, 223–232. [Электронный ресурс]/ Режим доступа: https://pdfs.semanticscholar.org/9234/6ec5276-203eb768cfa6d0a9bdaa49d4f060a.pdf(дата обращения 07.05.2017г)

33. Saiz A. & Simonsohn U. (2013). *Proxying for unobservable variables with Internet document frequency*. Journal of the European Economic Association, 11, 137–165 [Электронный ресурс]/ Режим доступа: https://www.deepdyve.-com/lp/wiley/proxying-for-unobservable-variables-with-internet-document-frequency-9iUgOuWarG(дата обращения 07.05.2017г)

34. Preis, T. Moat, H.S. & Stanley, H.E. (2013). *Quantifying Trading Behavior in Financial Markets Using Google Trends. Science Reports*. 3, 1684; DOI:10.1038/srep01684, 2013 [Электронный ресурс]/ Режим доступа: https://www.nature.com/articles/srep01684 (дата обращения 011.05.2017г)

Ссылки

1. http://www.searchengineshowdown.com/features/google/review.html
2. http://en.wikipedia.org/wiki/Web_crawler
3. http://www7.wwwconference.org/1921/com1921.htm
4. http://www.press.umich.edu/jep/07-01/bergman.com
5. http://en.wikipedia.org/wiki/Search_engine
6. http://www.robotstxt.org
7. https://moz.com/beginners-guide-to-seo/how-search-engines-operat
8. https://www.codementor.io/magarrent/how-to-install-laravel-5-xampp-windows-du107u9ji#install-xampp
9.https://confluence.jetbrains.com/display/PhpStorm/Composer+Support+in+PhpStorm

10. http://bourabai.ru/einf/chapter121.htm

ПРИЛОЖЕНИЕ

Программная реализация на языке PHP

Search.dev\app\repositories\se\repository.php

```
<?php
namespace App\Repositories\Se;
use GuzzleHttp\Client;
use GuzzleHttp\Cookie\FileCookieJar;
use GuzzleHttp\Promise;
class Repository
{
   protected $client;
   protected $cookie;
   protected $cookie_name = 'se.txt';
   protected $cookie_path;
   protected $user_agent = 'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/54.0.2840.87 Safari/537.36 OPR/41.0.2353.56';
   public function __construct()
   {
      $this->cookie_path = $this->getCookiePath();
      $this->cookie = $this->getCookie();
      $this->client = new Client([
         'cookies' => $this->cookie,
         'headers' => [
            'Accept-Encoding' => 'gzip, deflate, lzma, sdch',
            'Accept-Language' => 'ru-RU,ru;q=0.8,en-US;q=0.6,en;q=0.4',
            'DNT' => '1',
            'User-Agent' => $this->user_agent ]  ]);
   }
   public function __destruct()
   {
      $this->saveCookie();
   }
   protected function getCookiePath(): string
   {
      return storage_path('app') . '/' . $this->cookie_name;
   }
   protected function getCookie(): FileCookieJar
   {
      return new FileCookieJar($this->cookie_path);
   }
   protected function saveCookie()
   {
      $this->cookie->save($this->cookie_path);
   }
   protected function getContentFromBing($query)
   {
      $pages = [0, 10]; $promises = [];
```

```
        foreach ($pages as $page) {
            $promises[] = $this->client->getAsync('http://www.bing.com/search', [
                'headers' => [
                    CURLOPT_SSL_VERIFYPEER => FALSE,
                    CURLOPT_SSL_VERIFYHOST => FALSE,
                    CURLOPT_FTP_SSL => CURLFTPSSL_TRY
                ],
                'query' => [
                    'first' => $page * 10,
                    'format' => 'rss',
                    'q' => $query
                ]
            ]);
        }
        return Promise\settle($promises)->wait();
    }

    public function getFromSearchEngine($query)
    {
        $responses = $this->getContentFromBing($query);
        $results = [];
        foreach ($responses as $response) {
            $page = simplexml_load_string((string)$response['value']->getBody());
            foreach ($page->channel->item as $item) {
                $results[] = [
                    'description' => str_replace('...', '', (string)$item->description),
                    'link' => (string)$item->link,
                    'title' => (string)$item->title
                ];
            }
        }
        return $results;
    }
}
```

Search.dev\app\user.php

```
<?php
namespace App;
use Illuminate\Notifications\Notifiable;
use Illuminate\Foundation\Auth\User as Authenticatable;
class User extends Authenticatable
{
    use Notifiable;
    /**
     * The attributes that are mass assignable.
     *
     * @var array
     */
    protected $fillable = [
        'name', 'email', 'password',];

    /**
```

```
     * The attributes that should be hidden for arrays.
     *
     * @var array
     */
    protected $hidden = [
        'password', 'remember_token',
    ];
}
```

Search.dev\app\Http\controllers\controller.php

```
<?php
namespace App\Http\Controllers;
use Illuminate\Foundation\Bus\DispatchesJobs;
use Illuminate\Routing\Controller as BaseController;
use Illuminate\Foundation\Validation\ValidatesRequests;
use Illuminate\Foundation\Auth\Access\AuthorizesRequests;

class Controller extends BaseController
{
    use AuthorizesRequests, DispatchesJobs, ValidatesRequests;
}
```

Search.dev\app\Http\controllers\frontcontroller.php

```
<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
class FrontController extends Controller
{
    /**
     * Show the application dashboard.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        return view('index');
    }
}
```

Search.dev\app\Http\controllers\Api\controller.php

```
<?php
namespace App\Http\Controllers\Api;
use App\Http\Controllers\Controller as BaseController;
class Controller extends BaseController
{
}
```

Search.dev\app\Http\controllers\Api\Searchcontroller.php

```
<?php
namespace App\Http\Controllers\Api;
use App\Repositories\Se\Repository as Se;
use Illuminate\Http\Request;
class SearchController extends Controller
{
    protected $se;
```

```
    public function __construct(Se $se)
    {
        $this->se = $se;
    }
    public function index(Request $request)
    {
        $this->validate($request, [
            'query' => 'required|string'   ]);
        return response()->json($this->se->getFromSearchEngine($request->input('query')));
    }
}
```

Search.dev\app\Exceptions\handler.php

```
<?php

namespace App\Exceptions;

use Exception;
use Illuminate\Auth\AuthenticationException;
use Illuminate\Foundation\Exceptions\Handler as ExceptionHandler;

class Handler extends ExceptionHandler
{
    /**
     * A list of the exception types that should not be reported.
     *
     * @var array
     */
    protected $dontReport = [
        \Illuminate\Auth\AuthenticationException::class,
        \Illuminate\Auth\Access\AuthorizationException::class,
        \Symfony\Component\HttpKernel\Exception\HttpException::class,
        \Illuminate\Database\Eloquent\ModelNotFoundException::class,
        \Illuminate\Session\TokenMismatchException::class,
        \Illuminate\Validation\ValidationException::class,
    ];
    /**
     * Report or log an exception.
     *
     * This is a great spot to send exceptions to Sentry, Bugsnag, etc.
     *
     * @param  \Exception  $exception
     * @return void
     */
    public function report(Exception $exception)
    {
        parent::report($exception);
    }
    /**
     * Render an exception into an HTTP response.
     *
```

```
     * @param  \Illuminate\Http\Request  $request
     * @param  \Exception  $exception
     * @return \Illuminate\Http\Response
     */
    public function render($request, Exception $exception)
    {
        return parent::render($request, $exception);
    }
    /**
     * Convert an authentication exception into an unauthenticated response.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  \Illuminate\Auth\AuthenticationException  $exception
     * @return \Illuminate\Http\Response
     */
    protected function unauthenticated($request, AuthenticationException $exception)
    {
        if ($request->expectsJson()) {
            return response()->json(['error' => 'Unauthenticated.'], 401);
        }
        return redirect()->guest('login');
    }
}
```

Search.dev\app\console\kernel.php

```
<?php
namespace App\Console;
use Illuminate\Console\Scheduling\Schedule;
use Illuminate\Foundation\Console\Kernel as ConsoleKernel;
class Kernel extends ConsoleKernel
{
    /**
     * The Artisan commands provided by your application.
     *
     * @var array
     */
    protected $commands = [
        //
    ];
    /**
     * Define the application's command schedule.
     *
     * @param  \Illuminate\Console\Scheduling\Schedule  $schedule
     * @return void
     */
    protected function schedule(Schedule $schedule)
    {
        // $schedule->command('inspire')
        //          ->hourly();
    }
    /**
     * Register the Closure based commands for the application.
```

```
     *
     * @return void
     */
    protected function commands()
    {
        require base_path('routes/console.php');
    }
}
```

Search.dev\database\migration\create_users_table.php

```
<?php
use Illuminate\Support\Facades\Schema;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Database\Migrations\Migration;

class CreateUsersTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('users', function (Blueprint $table) {
            $table->increments('id');
            $table->string('name');
            $table->string('email')->unique();
            $table->string('password');
            $table->rememberToken();
            $table->timestamps();
        });
    }
    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::drop('users');
    }
}
```

Search.dev\database\factories\modelfactory.php

```
<?php
/*
|--------------------------------------------------------------------------
| Model Factories
|--------------------------------------------------------------------------
|
| Here you may define all of your model factories. Model factories give
| you a convenient way to create models for testing and seeding your
```

```
| database. Just tell the factory how a default model should look.
|
*/
$factory->define(App\User::class, function (Faker\Generator $faker) {
    static $password;
    return [
        'name' => $faker->name,
        'email' => $faker->safeEmail,
        'password' => $password ?: $password = bcrypt('secret'),
        'remember_token' => str_random(10),
    ];
});
```