Министерство образования Республики Беларусь

Учреждение образования

БелорусскиЙ государственный университет

информатики и радиоэлектроники

Факультет информационных технологий и управления

Кафедра интеллектуальных информационных технологий

**Отчет по преддипломной практике**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент группы 321702: | |  | | Пашкевич В.В. |
| Руководитель | |  | | Колб Д.Г. |
|  |  | |
|  |
|  |
|  |  | |
|  |  | |

Минск 2017

СОДЕРЖАНИЕ

[ПЕРЕЧЕНЬ УСЛОВНЫХ ОБОЗНАЧЕНИЙ 2](file:///F:\practice1.docx#_Toc482875638)

[ВВЕДЕНИЕ 3](file:///F:\practice1.docx#_Toc482875639)

[1 АНАЛИЗ ПОДХОДОВ К РЕШЕНИЮ ЗАДАЧ ДИПЛОМНОГО ПРОЕКТИРОВАНИЯ 5](file:///F:\practice1.docx#_Toc482875640)

[1.1 Анализ существующих аналогов 5](file:///F:\practice1.docx#_Toc482875641)

[1.1.1 livejournal.com 5](file:///F:\practice1.docx#_Toc482875642)

[1.1.2 bloger.by 7](file:///F:\practice1.docx#_Toc482875643)

[1.1.3 wix.com 8](file:///F:\practice1.docx#_Toc482875644)

[1.1.4 Сравнительный анализ аналогов 10](file:///F:\practice1.docx#_Toc482875645)

[1.2 Постановка требований к приложению 11](file:///F:\practice1.docx#_Toc482875646)

[1.3 Анализ подходов к решению задач 11](file:///F:\practice1.docx#_Toc482875647)

[1.4 Формулировка задачи дипломного проектирования 12](file:///F:\practice1.docx#_Toc482875648)

[2 ПРоектирование системЫ конструирования ВЕБ-блогов 13](file:///F:\practice1.docx#_Toc482875649)

[2.1 Проектирование структуры приложения 13](file:///F:\practice1.docx#_Toc482875650)

[3 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ 20](file:///F:\practice1.docx#_Toc482875651)

[3.1 Обоснование выбора технологий для разработки 20](file:///F:\practice1.docx#_Toc482875652)

[3.2 Обоснование выбора СУБД 22](file:///F:\practice1.docx#_Toc482875653)

[3.3 Разработка программной архитектуры 24](file:///F:\practice1.docx#_Toc482875654)

[3.4 Разработка приложения 25](file:///F:\practice1.docx#_Toc482875655)

[3.5 Организация взаимодействия с БД 27](file:///F:\practice1.docx#_Toc482875656)

[3.6 Работа с сущностями 28](file:///F:\practice1.docx#_Toc482875657)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 31](file:///F:\practice1.docx#_Toc482875658)

ПЕРЕЧЕНЬ УСЛОВНЫХ ОБОЗНАЧЕНИЙ

В настоящей пояснительной записке применяют следующие обозначения и сокращения:

WWW – World Wide Web;

HTTP – HyperText Transfer Protocol;

SQL – Structured Query Language;

БД – база данных;

ПО – программное обеспечение;

СУБД – система управления базами данных;

HTML – HyperText Markup Language;

JS – JavaScript;

AJAX – Asynchronous Javascript and XML;

jQuery – Javascript Query Language;

ORM – Object-Relational Mapping;

JPA – Java Persistence API;

DAO – Data Access Object;

DTO – Data Transfer Object;

ООП – Объектно-ориентированное программирование.

ВВЕДЕНИЕ

По данным последнего пресс-релиза Международного союза электросвязи (МСЭ) от 22.07.2016, количество пользователей интернета в мире составляет 3,5 миллиарда человек [1].

World Wide Web — глобальная компьютерная сеть на сегодняшний день содержит миллионы сайтов, на которых размещена всевозможная информация. Люди получают доступ к этой информации посредством использования технологии Internet. Для навигации в WWW используются специальные программы — Web-браузеры, которые существенно облегчают путешествие по бескрайним просторам WWW. Вся информация в Web-браузере отображается в виде Web-страниц, которые являются основным элементом WWW.

Каждый человек может пользоваться технологией Internet и просматривать, бескрайнее на данный момент, множество Web-страниц. Однако, что делать, если вы захотели делиться информацией в сети Internet? В этом случае можно обзавестись интернет-блогом. Блог является своеобразным дневников в сети, который доступен остальным пользователям сети.

Люди, которые ведут собственный блог, имеют возможность общаться с друзьями, живущими в других городах, заводить новые знакомства. Форма сообщений, действующая в блогах, делает её наиболее удобной, так как она не предполагает обязательной взаимности и других ограничений общения «один на один».

Человек, постоянно читающий блоги преследует следующие цели:

* получение информации;
* чтение и развлечение;
* отслеживание реакции публики на те или иные действия (в самом деле, блоги представляют собой готовую огромную фокус-группу);
* чтение ради социализации, ощущения себя причастным к жизни известных людей.

Но не каждый может создать свой собственный сайт для размещения веб-блогов. Данный проект предоставит возможность пользователям размещать свои собственные блоги, при этом не имея представления, о том, как это сделано внутри. Пользователи получат возможность создавать собственные блоги, размещая различного вида статьи. Каждый пользователь, читая блоги, сможет найти своих единомышленников по определенной теме. Находя тех, кто увлечен вашим хобби, можно получить что-то новое, либо же наоборот подсказать.

Рассмотрим более конкретно причины, которые делают блоги популярными:

* записывать нужную информацию в блог очень просто и удобно;
* скорость появления новостей. Например, новости в блогах могут появляться так же быстро, как в реальной жизни, но в блогах можно еще найти мнения других пользователей о данной новости и самому принять участие в обсуждении. При увеличении количества пользователей, разнообразие информации может быстро превзойти размеры различных энциклопедий;
* возможность заработка. Это может быть реклама, партнерские программы и т.п. Размер заработка непосредственно зависит от количества посетителей вашего блога;
* при помощи блогов можно познакомиться и пообщаться с довольно интересными людьми и неординарными личностями, обрести новое представление, об, казалось бы, уже изученной вещи.

Исходя из того, что количество пользователей интернета высоко, и веб-блоги обладают популярностью в современном обществе, можно сделать вывод, что разработка системы конструирования веб-блогов является актуальной.

Цель проекта: разработка системы, которая предоставит возможность создания веб-блогов.

Исходя из поставленной цели, были определены задачи:

* анализ подходов к решению задач дипломного проектирования
* проектирование системы конструирования веб-блогов;
* программная реализация;
* технико-экономическое обоснование эффективности разработки и реализации системы.

1 АНАЛИЗ ПОДХОДОВ К РЕШЕНИЮ ЗАДАЧ ДИПЛОМНОГО ПРОЕКТИРОВАНИЯ

**1.1 Анализ существующих аналогов**

На текущий момент на рынке предоставлено достаточное количество приложений, предоставляющих возможность вести блог. Рассмотрим некоторые из них более подробно.

1.1.1 livejournal.com

Рассмотрим один из самых популярных сервисов по созданию блогов – livejournal.com. На главной странице расположены, набирающие популярность, статьи (рис. 1.1).

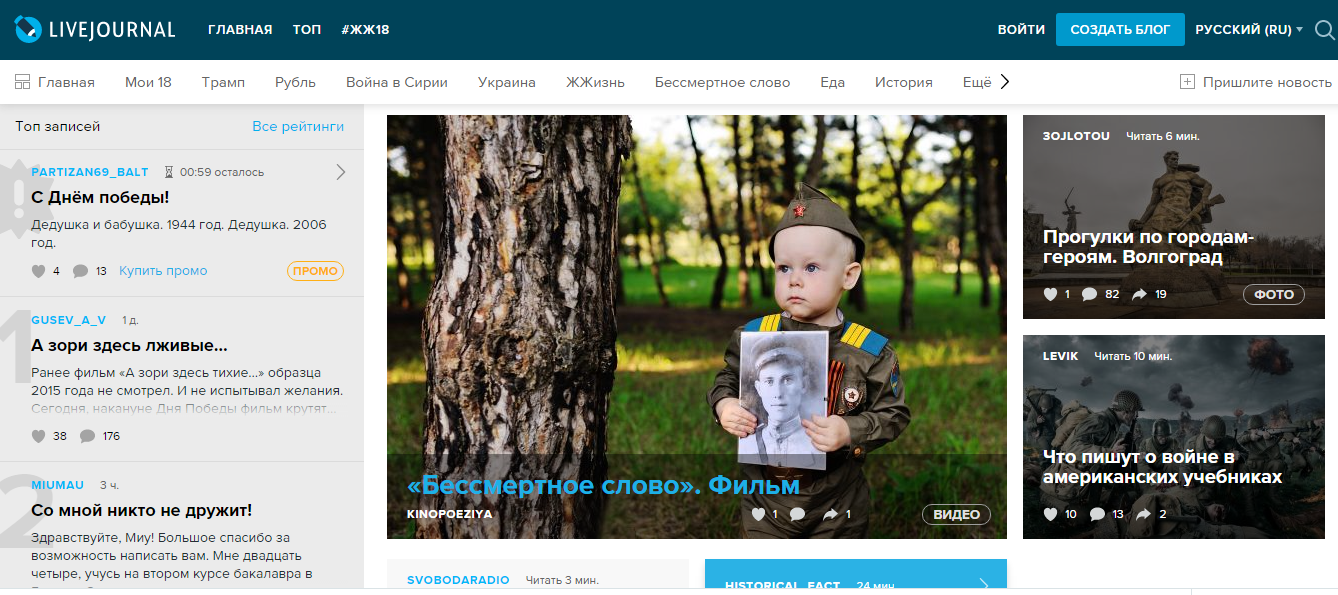


Рисунок 1.1 – Главная страница портала livejournal.com

На странице регистрации журнала необходимо указать имя пользователя, адрес email, пароль, дату рождения, пол, так же имеется reCAPTCHA от Google для защиты от ботов (рис 1.2).

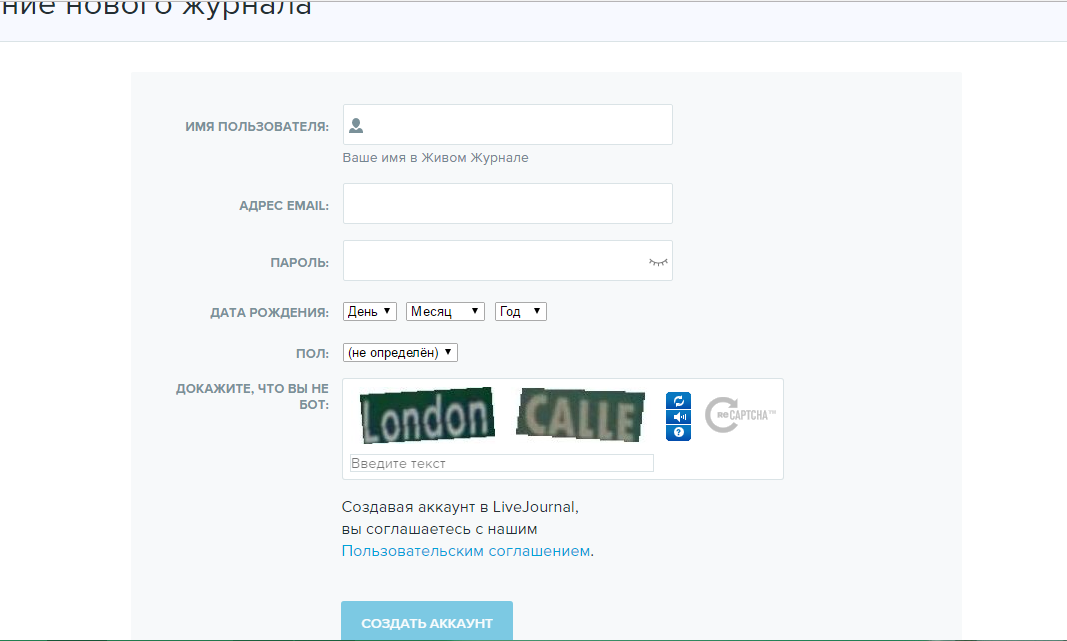


Рисунок 1.2 – Форма регистрации на портале livejournal.com

На портале есть множество сопутствующих возможностей помимо ведения блогов:

* кастомизация фона, текста;
* наличие платного контента;
* возможность добавление друзей (подписка на интересных вам блогеров);
* рейтинг записей, пользователей;
* возможность отправки сообщений.

Для написания записи предоставляются визуальный и html редакторы. Есть возможность выбора журнала для оставления записи. Присутствуют следующие возможности (рис 1.3, рис 1.4):

* добавления меток к записи;
* ограничение видимости записи;
* исключить запись из рейтинга;
* не оповещать друзей об данной записи.

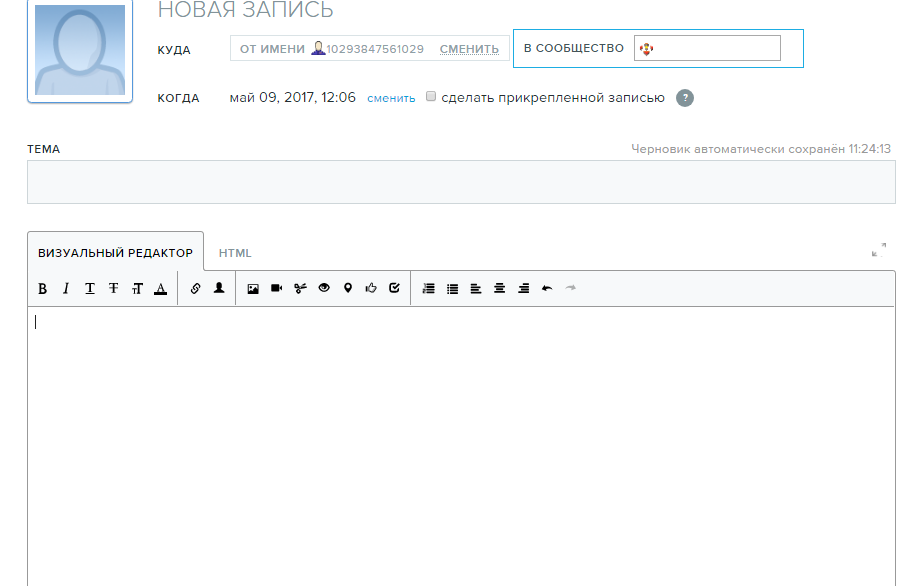


Рисунок 1.3 – Форма создания записи

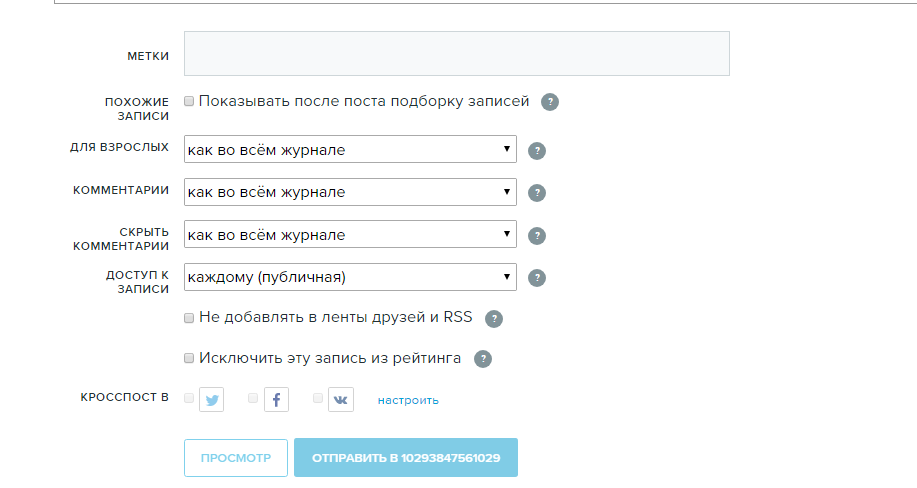


Рисунок 1.4 – Форма создания записи

К недостаткам данного портала следует отнести отсутствие возможности разграничивать статьи по темам, используя один аккаунт. Данная возможность предоставляется созданием еще одного журнала, для чего следует зарегистрироваться еще раз.

**1.1.2** **bloger.by**

Рассмотрим web-сайт, предоставляющий возможность вести собственные блоги bloger.by. Чтобы начать работать с блогом, необходимо зарегистрироваться. Регистрация на данном сайте проходит с указанием номера мобильного телефона и последующим подтверждением. Пройдя процесс регистрации, мы можем авторизоваться и попытаться оставить свой первый пост в блоге. При таком действии нам предоставляют следующую форму для создания поста (рис. 1.5).

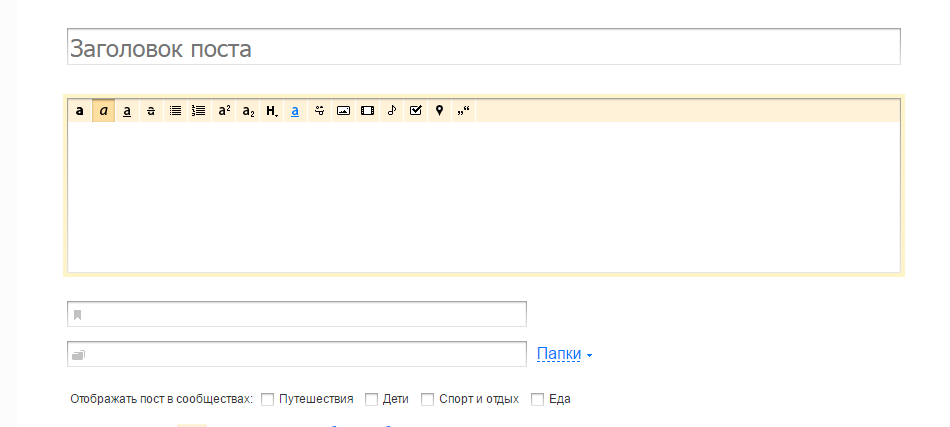


Рисунок 1.5 – Форма создания поста

Изображенные иконки предоставляют немалые возможности форматирования текста, вставки фотографии и т.д. Однако при первом использовании все те иконки кажутся незнакомыми и непонятно их предназначение. Для того чтобы вставить фотографию необходимо отыскать иконку фотографии и по щелчку откроется привычная форма для указания пути к файлу на вашем компьютере, либо другом устройстве.

К достоинствам этого приложения можно отнести следующие пункты:

* широкий спектр возможностей;
* возможность вести статистику.

Недостатки:

* несмотря на широкий спектр возможностей, отсутствует возможность вставки видео;
* интуитивно непонятный интерфейс для форматирования;
* излишняя безопасность с указанием мобильного телефона.

**1.1.3 wix.com**

Портал wix.com является невероятно мощным инструментов для создания красивых блогов. Попадая на главную страницу портала, мы наблюдаем хорошо продуманный дизайн (рис 1.6).

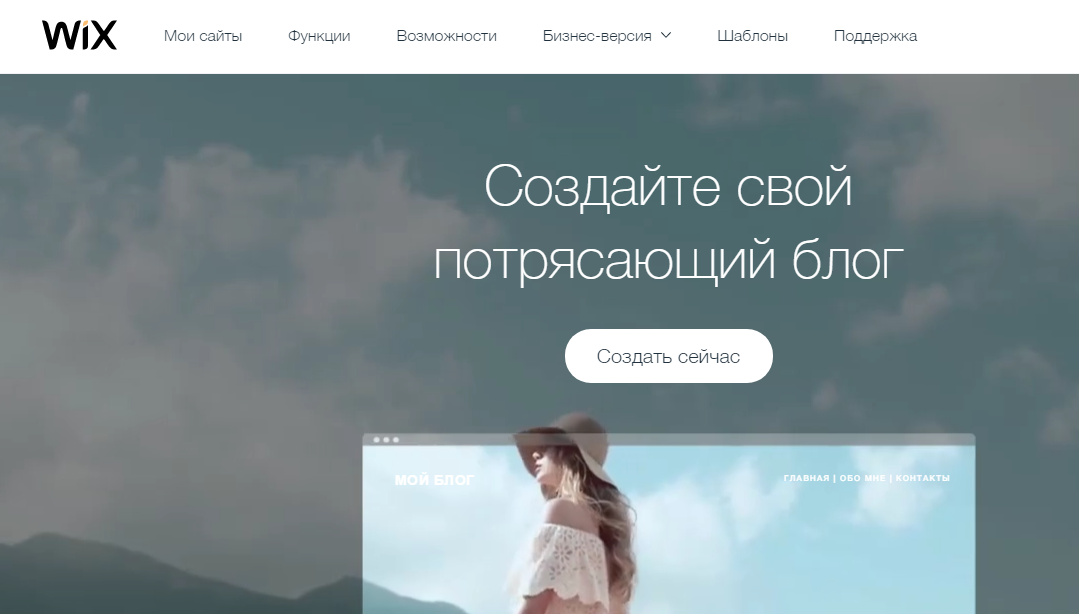


Рисунок 1.6 – Главная страницы портала wix.com

При непосредственном нажатии по кнопке “Создать сейчас” мы получаем огромный список шаблонов (рис 1.7).

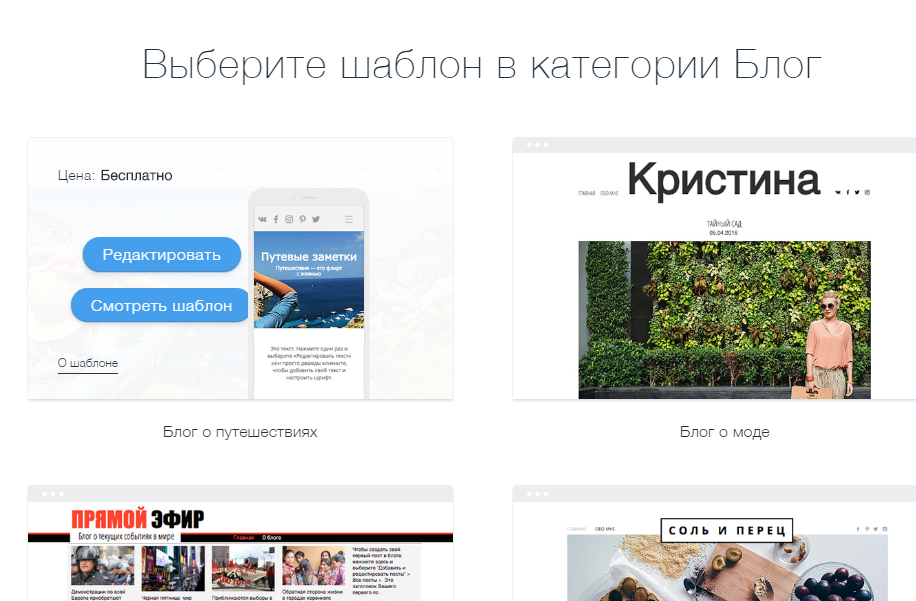


Рисунок 1.7 – Выбор шаблона на портале wix.com

При редактировании шаблона перед нами также открывается немалые возможности. Предоставляемые возможности чуть ли не достигают возможностей таких программ, как Adobe Photoshop. В итоге мы получаем чуть ли не персональный мини сайт. С различными встроенными ссылками и другими вещами.

К достоинствам этого приложения можно отнести следующие пункты:

* хорошо продуманный интерфейс;
* огромное количество инструментов;
* огромный выбор уже готовых шаблонов;

Недостатки:

* необходимо потратить время на изучение инструментов
* более подходит для тех, кто хочет создать красивый мини-сайт, а не блог. Например, для рекламы своего магазина. Тратя время на изучение инструментов, мы забыли об основных функция блогов – таких, как чтение информации, развлечение, обсуждение.

**1.1.4 Сравнительный анализ аналогов**

Рассмотренные порталы имеют различные функции, но объединены одной целью – распространение информации по сети интернет. Объем предоставляемых функций непосредственно зависит от сложности реализации.

Проведем сравнение рассмотренных порталов по некоторым критериям функционала. Сравнение представлено в таблице 1.1.

Таблица 1.1

|  |  |  |  |
| --- | --- | --- | --- |
| Критерий | bloger.by | wix.com | livejournal.com |
| Продуманный интерфейс | + | + | + |
| Возможность создания собственного блога | + | + | + |
| Простота использования | + | - | + |
| Интуитивно понятный интерфейс создания блога | - | - | + |
| Предоставление видео контента | - | + | + |
| Разграничение  контента, с помощью одного аккаунта | - | - | - |

По данной таблице видно, что каждый из порталов является довольно работоспособным. Однако не один из них не достигает высшей планки.

1.2 Постановка требований к приложению

Проведенный анализ показал, что на текущий момент имеет смысл в разработке приложения для конструирования блогов. Однако для достижения конкурентоспособности на рынке, необходимо превзойти существующие аналоги. Приложению установлены следующие требования:

* простота регистрации;
* отсутствие лишних функций;
* максимальная простота в навигации;
* максимально интуитивный интерфейс для наполнения статьи информацией;
* присутствие видео-контента;
* возможность отслеживать новые статьи интересующих пользователей;
* разграничение статей по темам, используя один аккаунт;
* возможность ведения рейтинга;
* возможность комментирования;
* возможность поиска.

Для работы приложения обязательным является подключение к сети интернет, наличие браузера для просмотра web-страниц.

1.3 Анализ подходов к решению задач

Исходя из поставленных требований, для реализации приложения необходимо решить несколько основных технических задач:

* для отслеживания новостей необходимо создать слушателя события;
* для обеспечения возможности поиска следует использовать готовый полнотекстовый движок;
* решить за счет каких технологий будет реализован интерфейс приложения.

Для отслеживания обновлений конкретного пользователя, необходимо создать слушателя события. Событием будет являться публикация новости пользователем. После публикации слушатель будет производить рассылку уведомлений по почтовым адресам подписчиков (пользователей, заинтересованных в этих новостях).

В качестве поиска разумнее всего будет использование одного из готовых движков, которых на данный момент предостаточно. Например, это может быть один из нижеперечисленных движков:

* Sprinx;
* Apache Lucene;
* Xapian;
* Hibernate search;
* ElasticSearch.

Одним из элементов удобства интерфейса будет являться навигационная панель. Навигационная панель веб-сайта — это область веб-страниц, на которой в некотором упорядоченном виде расположены ссылки на разделы и (или) страницы сайта, и единственная функция которой — предоставить пользователю удобное средство для перемещения по веб-сайту. Далее приведена классификация элементарных видов панелей навигации:

* одноуровневый список:
  + горизонтальный одноуровневый список;
  + вертикальный одноуровневый список;
* двухуровневый список:
  + двухуровневый список с фиксацией;
  + динамический двухуровневый список;
  + развернутый двухуровневый список;
  + полуразвернутый двухуровневый список.

1.4 Формулировка задачи дипломного проектирования

В ходе выполнения дипломного проекта необходимо разработать приложение, позволяющее пользователю конструировать собственные веб-блоги. В разрабатываемой системе необходимо достичь максимальной простоты использования и удобства.

2 ПРоектирование системЫ конструирования ВЕБ-блогов

2.1 Проектирование структуры приложения

При регистрации пользователь получает письмо. В письме содержится активационная ссылка, которая содержит ключ, сгенерированный для данного пользователя. Для каждого пользователя должен хранится только один ключ (тип «один-к-одному»). В случае отправки активационного письма повторно, предыдущий ключ удаляется, а после генерируется новый.

После регистрации пользователь может создавать блоги. Количество блогов неограниченно, что предоставляет возможность сортировки статей по темам, используя один аккаунт (тип «один-ко-многим»). Одному блогу может принадлежать множество статей (тип «один-ко-многим»). Для каждой статьи может быть изменен рейтинг (тип «один-ко-многим»). При изменении рейтинга сохраняются идентификаторы пользователя и статьи, что позволит избежать повторной оценки статьи. Также каждую статью можно комментировать (тип «один-ко-многим»).

Пользователь может подписаться на изменения понравившегося блогера. В данном случае следует использовать тип «многие-ко-многим». На рисунке 2.1 представлена диаграмма сущность-связь системы.

В системе будут использоваться три роли пользователей:

* неавторизованный;
* авторизованный;
* администратор.

Неавторизованный пользователь имеет доступ к просмотру существующих профилей, блогов, статей и отслеживать новые статьи на портале.

Для авторизации, необходимо зарегистрировать аккаунт, после чего перейти по активационной ссылке, указанной в электронном письме. В случае, если активационное письмо не получено, предусмотрена функция повторной отправки сообщения. В случае, если аккаунт активирован, но войти не удается, предусмотрена функция восстановления пароля. После входа в систему, возможности неавторизованного пользователя расширяются возможностями авторизованного.

Авторизованный пользователь имеет возможность создания собственного блога. Создав блог, открывается возможность для написания

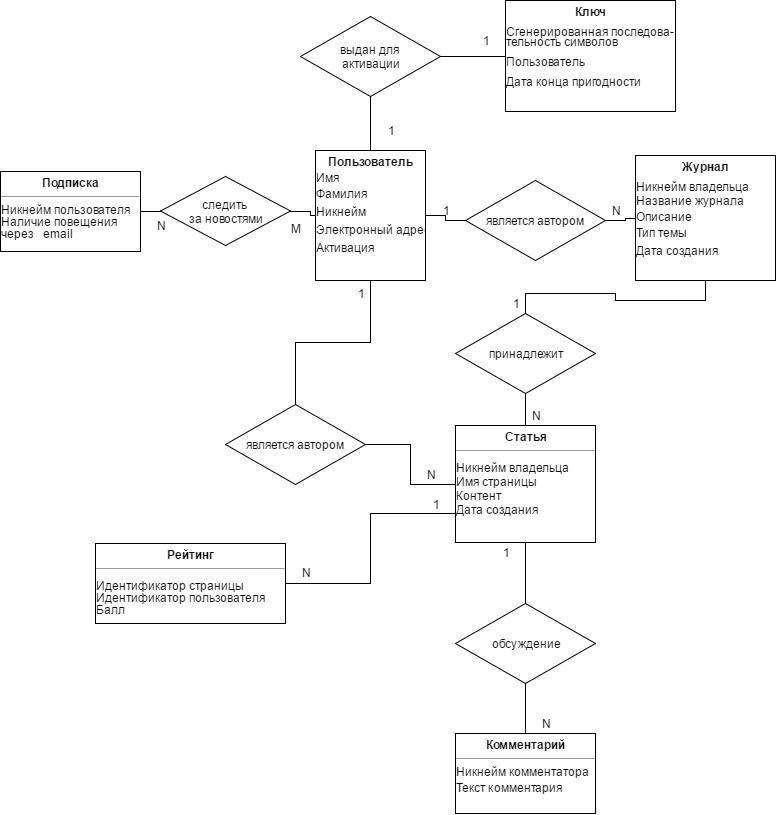


Рисунок 2.1 – Диаграмма «сущность-связь» предметной области

статей. Авторизованный пользователь может редактировать свой профиль, блог или статью. Также авторизованный пользователь имеет возможность отслеживать за новыми статьями у выбранного пользователя. Просматривая статьи, авторизованный пользователь имеет возможности оценить статью, оставить комментарий.

Администратору портала добавляются следующие возможности:

* редактирование статьи любого пользователя;
* удаление статьи или блога любого пользователя;
* редактирование профиля любого пользователя;
* удаление пользователя;
* блокировка пользователя.

На рисунке 2.2 представлена диаграмма вариантов использования системы.

Проектируемое приложение должно соответствовать принципам ООП. Следовательно, для написания приложения следует определиться с классами. Со стороны сервера все действия, которые совершает пользователь, являются HTTP-запросами. Необходимо выделить классы, которые будут заняты обработкой запросов. Данные классы будут носить общее название – контроллеры. При получении запроса, контроллер должен получить все параметры запроса (если они есть). Например, параметрами могут быть имя, фамилия и другие данные, передаваемые при подтверждении регистрации. Для данного приложения определено несколько контроллеров:

* AuthController. Данный контроллер будет отвечать на все запросы, связанные с регистрацией и авторизацией пользователей;
* UserController. Данный контроллер будет отвечать на все запросы, связанные с обработкой информацией пользователей. Например, создание журнала, статьи, редактирование профиля, взаимодействие между пользователями;
* AjaxController. Данный контроллер будет отвечать на все запросы, связанные с обработкой AJAX-запросов;
* HomeController. Данный контроллер будет отвечать на все запросы, связанные с обработкой главных страницы и других общих функций приложения.

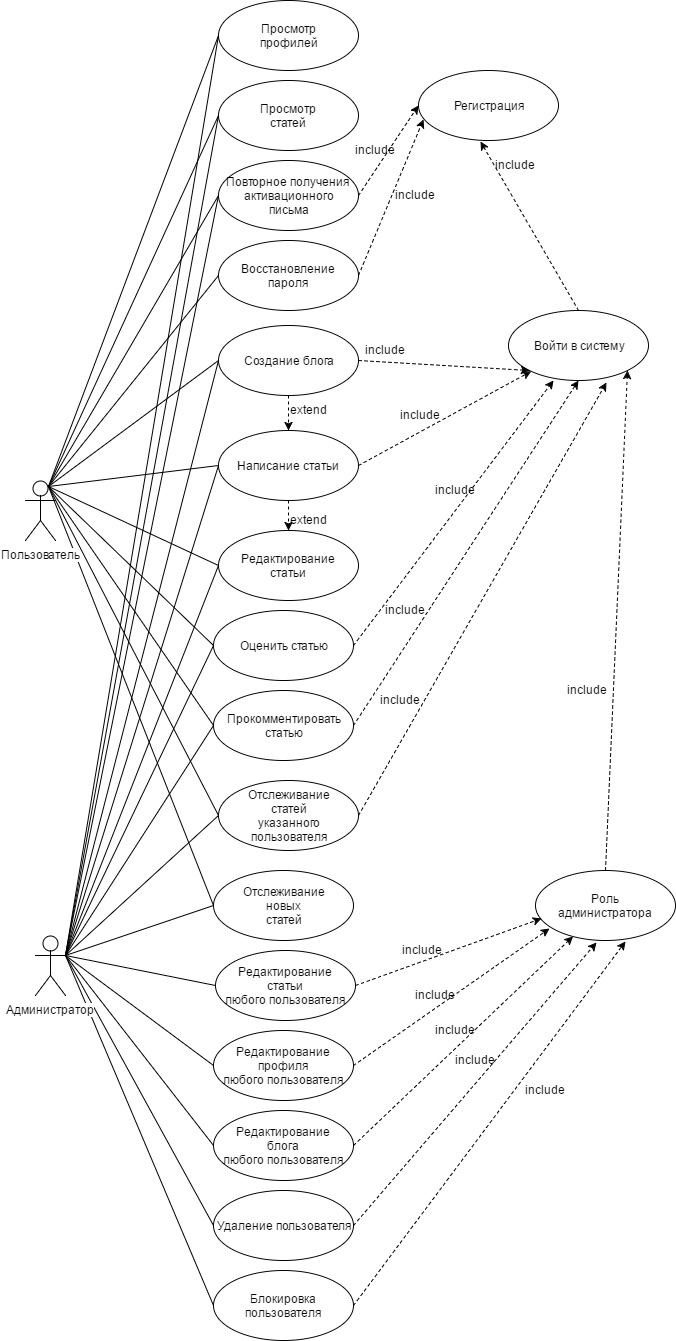


Рисунок 2.2 – Диаграмма вариантов использования Use Case

Однако, следуя принципам ООП, контроллеры не должны нести в себе логику. Их цель – принятие запроса, и передача данных на обработку. Обработкой данных будут заниматься классы, называемые сервисами. Также, следуя принципам ООП, для сервиса будет выделен интерфейс, который следует реализовать. Необходимо выделить некоторое количество сервисов:

* UserService. Работа данного сервиса направлена на большинство функций, связанных с пользователем;
* UserDetailsService. Работа данного сервиса направлена на работу с текущим пользователем, авторизованным в системе;
* SecurityService; Работа данного сервиса направлена на работу с функциями, обеспечивающих безопасность системы;
* PageService. Работа данного сервиса направлена на работу непосредственно со статьями пользователей.

На сервисах лежит большая работа, однако они не отвечают за работу с базой данных. Для работы с базой данных будут использоваться, так называемые, DAO-классы. Следуя, установленным нормам, для каждой сущности отведен свой DAO-класс, отвечающий за взаимодействие сущности и базы данных. Выделим необходимые DAO-классы:

* UserDao. Отвечает за взаимодействие сущности User и базой данных;
* BlogDao. Отвечает за взаимодействие сущности Blog и базой данных;
* PageDao. Отвечает за взаимодействие сущности Page и базой данных;
* TagDao. Отвечает за взаимодействие сущности Tag и базой данных;
* RateDao. Отвечает за взаимодействие сущности Rate и базой данных;
* TokenDao. Отвечает за взаимодействие сущности Token и базой данных;
* RoleDao. Отвечает за взаимодействие сущности Role и базой данных;

Исходя из приведенных данных можно привести пример создания статьи пользователем. Нажимая на кнопку создании статьи, браузер отправляет запрос, который получает UserController и возвращает HTML-страницу с формой для заполнения. Далее пользователь заполняет форму контентом и еще раз отправляет запрос на UserController. UserController, в свою очередь, извлекает все параметры и передает на обработку в PageService, который по окончании обработки, передаст сущность Page на сохранение в PageDao.

Также в приложении должен присутствовать класс, отвечающий за константы в приложении. Наличие такого класса позволит легко кастомизировать приложение, в случае необходимости. На рисунке 2.3 представлена диаграмма классов для системы.

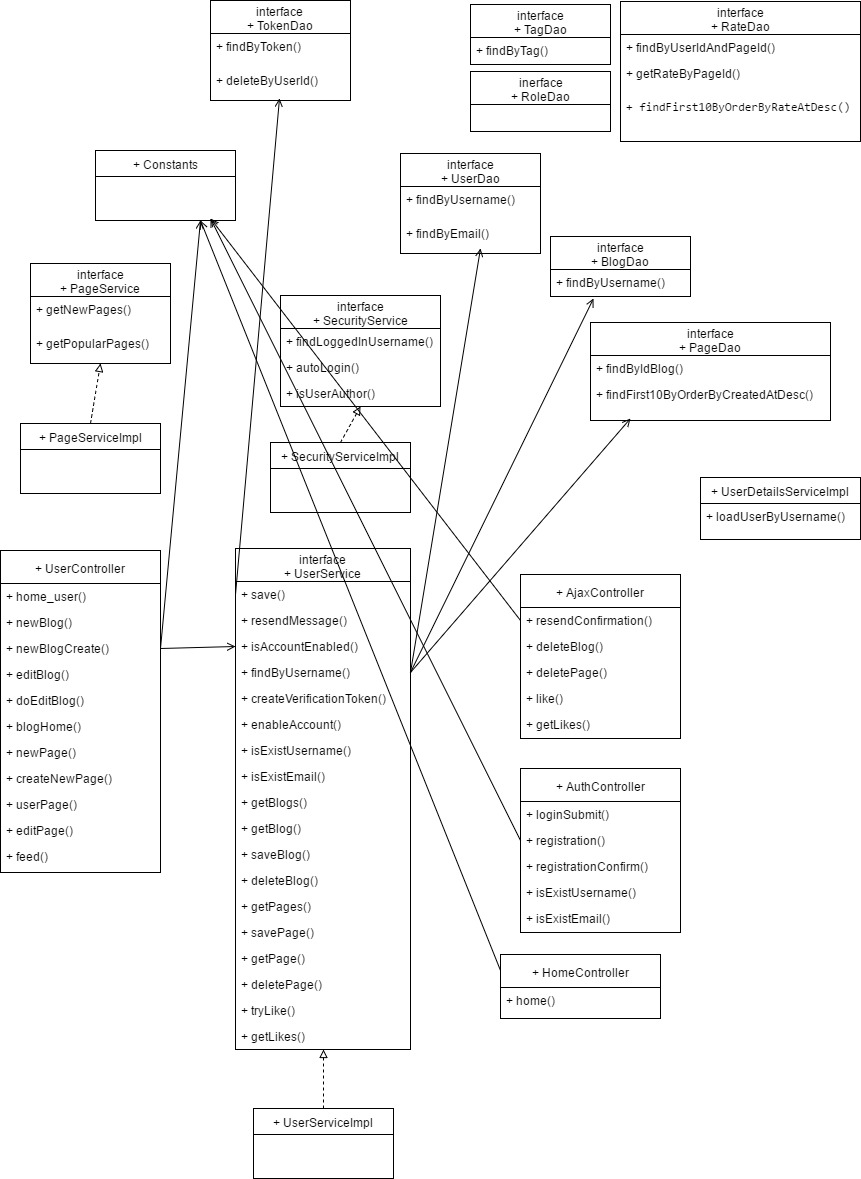


Рисунок 2.3 – Диаграмма классов

3 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ

3.1 Обоснование выбора технологий для разработки

Проект разрабатывается с использованием следующих инструментов:

* язык программирования Java;
* язык программирования JS;
* язык разметки HTML.

Язык программирования Java будет использован для написания

серверной части приложения. Рассмотрим некоторые преимущества этого языка:

* язык Java является объектно-ориентированным языком, что позволяет разбивать приложение на подпрограммы и использовать их по несколько раз;
* язык Java является кроссплатформенным. Данное свойство позволяет переносить приложение из одной системы в другую, что является довольно важным для web-приложения.
* надежность. Java обладает статической типизацией, обязательным контролем исключительных ситуаций, все это позволяет выявить множество ошибок;
* безопасность. Здесь отсутствуют механизмы обеспечивающие непосредственную работу с физической памятью (указатели и т.д.).
* наличие огромного сообщества разработчиков, в следствии чего наличие множества стандартных библиотек и продуманных решений;
* наличие мощных фрэймворков.

Для упрощения разработки серверной части будет использован фреймворк Spring Framework. Spring предоставляет разработчикам решения многих задач.

Spring - достаточно крупный фреймворк. Да нет, чего уж там - он просто огромный. Потому что создатели этого фреймворка ухитрились охватить практически все аспекты программирования промышленных Java-приложений. Соответственно, и составных частей у Spring Framework немало. Итак, вот они:

* IoC (Inversion of Control) контейнер;
* AOP-фреймворк (включая интеграцию с AspectJ);
* Data Access фреймворк;
* Transaction management;
* MVC-фреймворк;
* Remote Access фреймворк;
* Batch processing;
* Фреймворк аутентификации и авторизации;
* Remote Management;
* Messaging-фреймворк;
* Testing-фреймворк.

На иллюстрации вы можете увидеть схему фреймворка (рис.2.5). Из этой схемы видно, что первые два компонента в нашем списке являются самыми важными - это, в некотором роде, "сердце" фреймворка. Поэтому, прежде чем говорить дальше о каждом из перечисленных следом за этими двумя компонентов, давайте подробнее разберёмся с тем, что лежит в его основе - то есть, с шаблоном проектирования Inversion of Control и аспектно-ориентированным программированием [3].

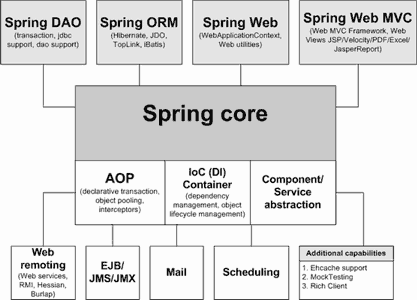


Рисунок 2.5 – Архитектура Spring Framework

Язык программирования Javascript используется для написания клиентской части приложения.

JavaScript изначально создавался для того, чтобы сделать web-странички «живыми». Программы на этом языке называются скриптами. В браузере они подключаются напрямую к HTML и, как только загружается страничка – тут же выполняются. JavaScript может выполняться не только в браузере, а где угодно, нужна лишь специальная программа – [интерпретатор](http://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BF%D1%80%D0%B5%D1%82%D0%B0%D1%82%D0%BE%D1%80). Процесс выполнения скрипта называют «интерпретацией». Современный JavaScript – это «безопасный» язык программирования общего назначения. Он не предоставляет низкоуровневых средств работы с памятью, процессором, так как изначально был ориентирован на браузеры, в которых это не требуется.

Что же касается остальных возможностей – они зависят от окружения, в котором запущен JavaScript. В браузере JavaScript умеет делать всё, что относится к манипуляции со страницей, взаимодействию с посетителем и, в какой-то мере, с сервером:

* создавать новые HTML-теги, удалять существующие, менять стили элементов, прятать, показывать элементы и т.п.;
* реагировать на действия посетителя, обрабатывать клики мыши, перемещения курсора, нажатия на клавиатуру и т.п.;
* посылать запросы на сервер и загружать данные без перезагрузки страницы (эта технология называется "AJAX");
* получать и устанавливать cookie, запрашивать данные, выводить сообщения [4].

Для удобства к дополнению Javascript, будет использован фреймворк jQuery. Фреймворк jQuery способствует более простой навигации и взаимодействию по элементам HTML-документов.

3.2 Обоснование выбора СУБД

Для работы с БД используется СУБД MySQL. Рассмотрим более подробно данную СУБД и определим причины, по которым именно она стала основой для данного проекта.

СУБД MySQL поддерживает SQL, структурированный язык запросов, и может применяться в качестве SQL-сервера. Это означает, что общаться с сервером можно на языке SQL: клиент посылает серверу запрос, тот его обрабатывает и отдает клиенту только те данные, которые были получены в результате этого запроса. Тем самым клиенту не требуется выкачивать данные и производить вычисления, как, например, в Microsoft Access.

Кроме того, MySQL – это ПО с открытым кодом, т.е. его можно свободно изучать и изменять.

С появлением Интернет-технологий, позволяющих создавать динамичные Web–страницы, необычайно возрос спрос и на СУБД, которые наиболее полно подходили бы для этого по быстродействию, надежности и стабильности. И здесь хорошо проявил себя пакет MySQL, который получился быстрым, простым и надежным, но, правда, за счет ухудшения функциональности. Среди недостатков данной СУБД: отсутствие транзакций, триггеров, хранимых процедур, вложенных запросов.

Перечислим преимущества СУБД MYSQL:

* многопоточность, поддержка нескольких одновременных запросов;
* оптимизация связей с присоединением многих данных за один проход;
* записи фиксированной и переменной длины;
* ODBC драйвер;
* гибкая система привилегий и паролей;
* гибкая поддержка форматов чисел, строк переменной длины и меток времени;
* быстрая работа, масштабируемость;
* совместимость с ANSI SQL;
* бесплатна в большинстве случаев;
* хорошая поддержка со стороны провайдеров услуг хостинга;
* быстрая поддержка транзакций через механизм InnoDB.

Важным преимуществом данной СУБД является её сообщество. Как следствие открытости кода, бесплатности программы, стабильной и надежной ее работы образовалось сообщество людей, которые не просто лояльны к MySQL, но и всячески участвуют как в развитии самого пакета, так и в обучении менее опытных людей работе с ним. Существует огромное количество листов рассылки и конференций, где можно получить бесплатную помощь в любое время суток.

MySQL разрабатывает впечатляющие современные средства для реальных заказчиков. Эта компания первой реализовала раздел LIMIT в языке [SQL](http://technet.microsoft.com/ru-ru/sqlserver/default.aspx). MySQL является первопроходцем в областях полнотекстового поиска и репликации. Им создана уникальная двухзвенная архитектура подключаемых подсистем управления хранением данных (Pluggable Storage Engine Architecture). MySQL является пионерами в областях scale-out и использования кластеров для управления базами данных в основной памяти.

Так что в этом смысле они выполняют передовую исследовательскую работу. Но целью компании всегда является создание чего-либо, что может быть быстро проверено реальными пользователями.

По всем этим причинам MySQL стала незыблемым стандартом в области СУБД для web, а теперь в ней развиваются возможности для использования ее в любых критичных бизнес-приложениях, то есть конкурирует на равных с такими СУБД таких производителей, как Oracle, IBM, Microsoft и Sybase.

3.3 Разработка программной архитектуры

Используя фреймворк Spring для реализации, будет использована существующая архитектура для обработки запросов. Данная архитектура изображена на рисунке 3.1.

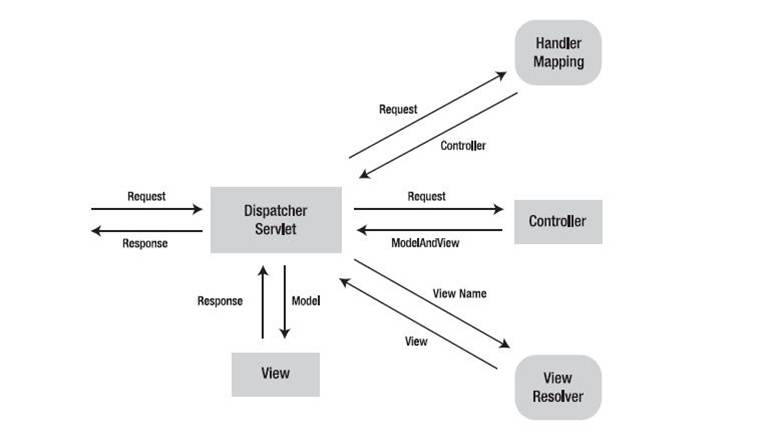


Рисунок 3.1 – Алгоритм обработки запросов

Для повышения возможностей в разработке проекта будет использован фреймворк Apache Maven. Apache Maven – [фреймворк](https://ru.wikipedia.org/wiki/%D0%A4%D1%80%D0%B5%D0%B9%D0%BC%D0%B2%D0%BE%D1%80%D0%BA) для [автоматизации сборки](https://ru.wikipedia.org/wiki/%D0%90%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F_%D1%81%D0%B1%D0%BE%D1%80%D0%BA%D0%B8) проектов на основе описания их структуры в файлах на языке POM([Project Object Model](https://en.wikipedia.org/wiki/Project_Object_Model)), являющемся подмножеством [XML](https://ru.wikipedia.org/wiki/XML) [[8]](https://ru.wikipedia.org/wiki/Apache_Maven#cite_note-3). С помощью фреймворка Maven будет возможность легко добавлять в проект необходимые зависимости, помещая их в pom.xml. Для подключения зависимости необходимо заполнить 3 поля:

* group id;
* artifact id;
* version.

Например, так будет выглядеть зависимость для Spring Boot:

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

<version>1.5.3.RELEASE</version>

</dependency>

Все классы созданные в процессе разработки проекта располагаются, в отведенных по предназначению для них, пакетах. Отдельно стоит выделить пакет, предназначенный для констант, который позволяет кастомизировать проект. Все статические файлы (\*.js, \*.css) располагаются в директории /resources/static.

В процессе регистрации пользователю отправляется письмо на электронный адрес, данной действие производится в отдельном потоке, для уменьшения времени ожидания пользователя.

3.4 Разработка приложения

И так, мы выбрали все необходимые технологии для разработки приложения. Для начала необходимо создать Maven project. На рисунке 3.2 изображена типовая структура созданного приложения.

├── pom.xml

└── src

└── main

├── java

└── resources

Рисунок 3.2 – Типовая структура Maven-проекта

Необходимо внести все зависимости проекта в pom.xml. Нам потребуется:

* Spring Boot Starter Parent;
* Spring Boot Starter Data Jpa;
* Spring Boot Starter Security;
* Spring Boot Starter Thymeleaf;
* Thymeleaf Extras Springsecurity4;
* Mysql connector java;
* Spring Boot Starter Mail.

Далее требуется разбить приложения на необходимые пакеты:

* constants. Пакет, предназначенный для содержания класса с константами;
* dao. Пакет, предназначенный для содержания DAO-классов приложения;
* listeners. Пакет, предназначенный для содержания слушателей событий приложения;
* model. Пакет, предназначенный для содержания сущностей приложения;
* service. Пакет, предназначенный для содержания сервис классов приложения;
* spring. Пакет, предназначенный для содержания конфигураций, необходимых для корректной работы Spring-фрэймворка;
* utils. Пакет, предназначенный для содержания различных статический вспомогательных функций;
* web. Данный пакет будет включать пакеты controllers (пакет, для контроллеров приложения) и dto (пакет для DTO-объектов приложения).

Диаграмма классов для приложения предоставлена на рисунке 3.3.

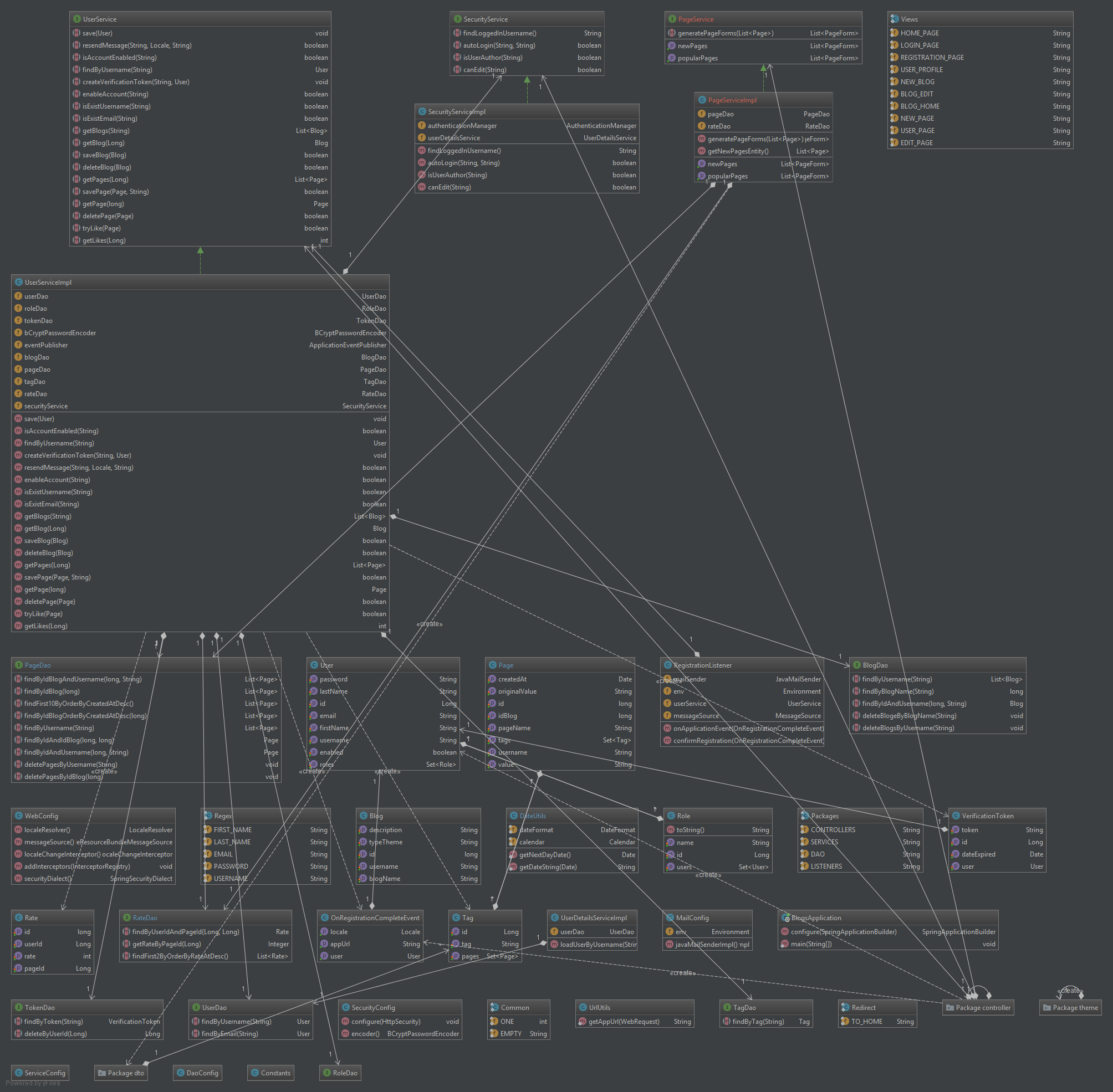


Рисунок 3.3 – Диаграмма классов приложения

3.5 Организация взаимодействия с БД

Spring, как и большинство фреймворков в наши дни, предлагает объектно-реляционное отображение (ORM) системы. ORM помогает вывести вас из бизнеса написания SQL и позволяют управлять базой данных используя язык программирования, в данном случае, Java.

Иcпользование реляционной базы данных для хранения объектно-ориентированных данных приводит к семантическому разрыву, заставляя программистов писать программное обеcпечение, которое должно уметь как обрабатывать данные в объектно-ориентированном виде, так и уметь сохранить эти данные в реляционной форме. Эта постоянная необходимость в преобразовании между двумя разными формами данных не только сильно снижает производительность, но и создает трудности для программистов, так как обе формы данных накладывают ограничения друг на друга [7].

Некоторые реализации ORM автоматически синхронизируют загруженные в память объекты с базой данных. Для того чтобы это было возможным, после создания объект-в-SQL-преобразующего SQL-запроса (класса, реализующего связь с БД) полученные данные копируются в поля объекта, как во всех других реализациях ORM. После этого объект должен следить за изменениями этих значений и записывать их в базу данных [7].

Системы управления реляционными базами данных показывают хорошую производительность на глобальных запросах, которые затрагивают большой участок базы данных, но объектно-ориентированный доступ более эффективен при работе с малыми объёмами данных, так как это позволяет сократить семантический провал между объектной и реляционной формами данных [7].

В качестве ORM можно использовать Hibernate, Java Persistence API (JPA) and Java Data Objects (JDO). Все из перечисленных ORM хорошо интегрированны с фреймворком Spring. Для разработки данной системы используется JPA. Для того, чтобы использовать JPA, необходимо добавить зависимость в pom.xml. Далее необходимо создать собственный класс, который будет унаследован от класса JpaRepository. Теперь мы можем описывать простые прототипы функций и JPA будет автоматически обрабатывать имя функции. Например, функция List<Blog> findByUsername(String username), объявленная в классе-наследнике JpaRepository<Blog, Long>, будет соответствовать нативному SQL-запросу 'SELECT \* FROM `table` WHERE `username` = “username”’, где table – имя таблицы для сущности Blog.

3.6 Работа с сущностями

Все определенные сущности, имеют аннотации @Entity, @Table, полученные из пакета javax.persistence.\*.

При регистрации пользователь должен заполнить предложенные поля. Имеет целесообразность валидация полей, предоставленных для заполнения пользователю. Для данных целей испозуется шаблон проектирования DTO. Для реализации шаблона создается дополнительный класс, дублирующий необходимые поля класса сущности. Далее мы можем воспользоваться существующими аннотациями для валидации: @Min, @Max, @NotNull и т.д. Также есть возможность создания собственной аннотации для валидации, реализуя интерфейс Spring Validator. В данном проекте используется существующая аннотация @Pattern, позволяющая подвергать поле на соответствие указанному паттерну регулярного выражения.

**ЗАКЛЮЧЕНИЕ**

В ходе преддипломной практики был ознакомлен с принципами охраны труда и техникой безопасности принятыми на предприятии.

Ознакомлен с нормами и правилами оформления программных и текстовых документов, используемых в дипломном проектировании.

Выполнен анализ предметной области, рассмотрены средства проектирования, произведено сравнение аналогов разрабатываемого приложения, поставлена цель и задача дипломного проектирования, выдвинуты требования к проектируемому продукту.

Исходя из всех полученных данных, были разработаны сценарий функционирования приложения и разработана база данных для приложения.

# СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

[1] Интернет в России и в Мире [Электронный ресурс]. – Режим доступа: http://www.bizhit.ru/

[2] TexTerra [Электронный ресурс]. – Режим доступа: https://texterra.ru/

[3] High-Tech Club [Электронный ресурс]. – Режим доступа: <https://www.kv.by>

[4] Современный учебник Javascript [Электронный ресурс]. – Режим доступа: <https://learn.javascript.ru/>

[5] Spring [Электронный ресурс]. – Режим доступа: https://spring.io/

[6] Tutorials for Java Concurrency [Электронный ресурс]. – Режим доступа: <https://www.tutorialspoint.com>

[7] Spring Docs [Электронный ресурс]. – Режим доступа: <https://docs.spring.io/>

[8] Apache Maven Project [Электронный ресурс]. – Режим доступа: <https://maven.apache.org/>

[9] Библиотека ресурсов интернет идустрии [Электронный ресурс]. – Режим доступа: http://www.i2r.ru