ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
МОРДОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ им. Н.П.
ОГАРЁВА»**

Факультет математики и информационных технологий
Кафедра фундаментальной информатики

УТВЕРЖДАЮ
Зав. кафедрой
канд. физ.-мат. наук, доцент
_____ А.Г. Смольянов
(подпись)
«10» 06 2019 г.

БАКАЛАВРСКАЯ РАБОТА

СОЗДАНИЕ СТАТИЧЕСКИХ ВЕБ-САЙТОВ С ПОМОЩЬЮ ГЕНЕРАТОРА JEKYLL

Автор бакалаврской работы _____ 10.06.19 _____ А. Ю. Солдаткин
подпись, дата

Обозначение бакалаврской работы БР-02069964-02.03.02-13-19

Направление 02.03.02 Фундаментальная информатика и информационные технологии

Руководитель работы
канд. физ.-мат. наук _____ 10.06.19 _____ А. В. Попов
подпись, дата

Нормоконтролер
канд. техн. наук, доцент _____ 10.06.19 _____ С. В. Гарина
подпись, дата

Рецензент
канд. физ.-мат. наук, доцент _____ 10.06.19 г. _____ Л. А. Сухарев
подпись, дата

Саранск

2019

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ МОРДОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ им. Н.П. ОГАРЁВА»

Факультет математики и информационных технологий
Кафедра фундаментальной информатики

УТВЕРЖДАЮ
Зав. кафедрой
канд. физ.-мат. наук, доц.
_____ А.Г. Смольянов
(подпись)
«12» 12 2018 г.

ЗАДАНИЕ НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
(в форме бакалаврской работы)

Студент Солдаткин Андрей Юрьевич

1 Тема «Создание статических веб-сайтов с помощью генератора Jekyll»

Утверждена приказом № 1028Z-С от 12.12.18

2 Срок предоставления работы к защите 13.06.19

3 Исходные данные для научного исследования литература и электронные источники по вопросам разработки веб-сайтов

4 Содержание выпускной квалификационной работы

4.1 Виды сайтов

4.2 Генераторы статических сайтов

4.3 Пример создания сайта на Jekyll

5 Приложения Код основных элементов веб-сайта

Руководитель работы
канд. физ.-мат. наук _____ 12.12.18 А. В. Попов
подпись, дата

Задание принял к исполнению _____ 12.12.18 А. Ю. Солдаткин
подпись, дата

РЕФЕРАТ

Бакалаврская работа содержит 54 страницы, 17 рисунков, 15 использованных источников, 1 приложение.

JEKYLL, HTML, CSS, GITHUB, ГЕНЕРАТОР, ВЕБ-САЙТ, ИНТЕРНЕТ, ХОСТИНГ, LIQUID.

Объектом исследования является процесс разработки статического веб-сайта с помощью генератора Jekyll.

Цель работы – рассмотрение технологических особенностей создания веб-сайта с использованием генератора статических сайтов, реализация веб-сайта и дополнительных возможностей.

В результате проведенной работы были изучены технологии, применяемые в настоящее время для разработки статических веб-сайтов, проанализированы особенности и возможности популярных генераторов статических веб-сайтов, статических и динамических сайтов, способы размещения веб-сайтов в сети Интернет. Разработан статический веб-сайт и дополнительные возможности «Поделится» и обратная связь.

Степень внедрения – частичная.

Область применения – сеть интернет.

Эффективность – высокая безопасность и скорость загрузки веб-сайта, очень низкая стоимость поддержки сайта.

Содержание

ВВЕДЕНИЕ

Интернет развивается стремительно и уже давно стал неотъемлемой частью нашей цивилизации. Глобальная сеть связывает практически все крупные научные и правительственные организации мира, университеты и бизнес-центры, социальные сети и онлайн магазины, образуя огромную базу данных информации. Быстро растет количество интернет ресурсов, а вместе с этим и количество пользователей. При таком большом количестве посетителей возникает проблема с перегрузкой веб-сайтов.

Много лет назад первопроходцы веб-программирования создавали свои первые сайты в блокноте и загружали их на хостинг по протоколу FTP. Позднее стало очевидно, что из статичных страниц нельзя создать что-то более масштабное, чем домашняя страничка. Для создания серьезных проектов начали использовать динамические сайты и серверные языки программирования (PHP, Python, Ruby, Java). Однако с новыми технологиями пришли и новые проблемы: низкая скорость загрузки контента и недоступность веб-ресурса при большом количестве посетителей. Поэтому сейчас для некоторых задач вновь начинают набирать популярность статические сайты.

Основное достоинство статических сайтов – это отсутствие задержки для генерации контента. Страницы на таких сайтах уже готовы, они единовременно передаются в память для мгновенной демонстрации пользователю, в том числе в системах с высокой нагрузкой.

Содержимое статических сайтов автоматически генерируется специальными программами-движками. Они позволяют создавать веб-проекты, основанные на статичных страницах, которые можно размещать на любом веб-хостинге, в том числе на GitHub.

Целью данной бакалаврской работы является разработка и размещение на сервисе GitHub статического сайта для ведения новостного блога. В соответствии с целью были сформулированы следующие задачи:

- провести анализ предметной области, сравнить возможности статических и динамических сайтов;

- рассмотреть популярные генераторы статических статических сайтов;

- разработать на базе одного из генераторов статический сайт для новостного блога;

- разместить сайт на бесплатном хостинге GitHub Pages и подключить к нему дополнительные возможности по комментированию записей и распространению ссылок в социальных сетях.

Бакалаврская работа состоит из введения, трех глав и заключения.

В первой главе рассмотрены основные типы веб-сайтов, описаны их принципы функционирования, а также проведено сравнение статических и динамических сайтов.

Во втором разделе рассмотрены современные технологии для генерации статических веб-сайтов, выбраны языки и фреймворки, которые будут использованы в дальнейшем при разработке сайта.

В третьем разделе описан процесс практической разработки статического веб-сайта с помощью генератора Jekyll и публикации этого сайта на хостинге Github Pages, а также рассмотрены варианты добавления статическим сайтам интерактивных возможностей с помощью сторонних сервисов.

1 Виды сайтов

Все сайты во всемирной паутине можно разделить на две большие группы: статические и динамические.

К статическим относятся сайты, состоящие из готовых HTML-страниц, готовых к использованию без дополнительной обработки.

Динамическими являются сайты, на которых контент формируется в зависимости от действий пользователя, то есть на основе информации из базы данных, которую запрашивают клиенты.

Рассмотрим более подробно особенности сайтов обоих типов, чтобы лучше выявить их достоинства и недостатки, а также понять, сайты какого типа лучше выбрать при разработке того или иного проекта [10].

1.1 Статические сайты

Статический сайт – это сетевой ресурс, состоящий из постоянных HTML-страниц, которые связываются гиперссылками в единое целое. Сами страницы могут содержать в себе информацию разного рода: текст, фотографии, видео, сценарии на языке JavaScript и т.д. Для корректировки статичных страниц разработчик должен внести непосредственные изменения в их HTML-код. Статические сайты получили свое название за то, что данные, которые в них содержатся, остаются практически неизменными. Таким образом, каждому пользователю, обращающемуся к странице, будет выведена в браузере одна и та же информация.

Каждая страница статического сайта верстается вручную или генерируется на основании определенного шаблона. Страницы хранятся на веб-сервере в формате HTML файлов. Обычно статический сайт содержит небольшое количество страниц.

Подобные сайты хорошо подходят для разработки сайтов-визиток какой-либо компании или частного лица. На таких сайтах не используется база

данных, поэтому по умолчанию на них отсутствует возможность регистрации пользователей и написания ими отзывов.

Редактировать статический сайт сможет только человек, разбирающийся в веб-разработке, потому что в нем нет административной панели и все обновления в проекте придется делать через его исходный код (HTML, CSS, JavaScript).

Итак, можно выделить следующие достоинства статических сайтов:

1. **Простота создания**. Статические сайты создаются быстро и просто, в сети можно найти много бесплатных заготовок под разные задачи.
2. **Высокая скорость работы**. Страницы статических сайтов сформированы заранее и каждый раз загружаются в неизменном формате, при этом нет необходимости делать запросы к базе данных. Это положительно сказывается на скорости загрузки.
3. **Дешевый или даже бесплатный хостинг**. Сайты этого типа занимают минимальный объем на сервере и практически не оказывают нагрузки на сервер. Это значит, что владельцам таких сайтов нужен лишь основной пакет услуг, без дополнительного места для баз данных.
4. **Простая смена хостинга**. Если владелец решит перенести сайт на другой хостинг, то сделать это будет очень просто – достаточно просто скопировать все файлы.

К недостаткам классических статических сайтов, которые верстаются вручную, можно отнести следующее:

1. **Сложно изменять содержимое страниц**. Для редактирования сайта требуется опыт веб-разработки, знакомство с разметкой HTML, стилями CSS и языком JavaScript.
2. **Проблемы с продвижением сайта**. Контент на статических сайтах обновляется редко, поэтому новых посетителей привлечь на сайт сложно.

3. **Ограниченная функциональность, неудобно использовать в больших проектах**.

Предположим, что мы хотим сделать с помощью статических страниц интернет-магазин. В этом случае при добавлении нового товара в каталог нужно будет вручную отредактировать код в нескольких страницах (как минимум это каталог товара и форма заказа). Кроме того, учет наличия товара потребуется вести с помощью какой-то сторонней программы.

Однако статические сайты продолжают использоваться по сей день. Дело в том, что одна из основных проблем, связанная со сложным процессом изменения контента, решается с помощью различных генераторов сайтов. Подробнее генераторы сайтов рассматриваются во второй главе.

1.2 Динамические сайты

Динамический сайт – это сетевой ресурс, состоящий из шаблонов и скриптов, которых хранятся и исполняются на сервере. Вся используемая на сайте информация обычно хранится в базе данных. Когда пользователь делает запрос страницы, соответствующая информация извлекается из базы данных, переносится в шаблон, образуя веб-страницу и пересылается веб-сервером в браузер пользователя.

Для отображения однотипных страниц используется страница-шаблон, в которую подгружается сформированный контент. При редактировании такой страницы-шаблона корректируется внешний вид множества страниц сайта.

Для обновления содержимого динамического сайта можно использовать различные систем управления контентом сайта (Content Management System, CMS). CMS позволяет поддерживать информационные веб-сайты различной сложности. Например, если есть необходимость добавить новый товар в каталог интернет-магазина, то сделать это гораздо проще, чем на статическом сайте. С помощью CMS в специальном поле заполняется информация о товаре,

после чего эта информация записывается в базу данных. Далее сайт будет просто загружать дополнительный товар и проблем с отображением информации не возникнет.

Достоинства динамических сайтов:

1. **Простота работы с CMS**. Для работы с CMS не нужно обладать специальными знаниями веб-разработчика (HTML, CSS, JavaScript, PHP). Операции добавления, изменения и удаления информации выполняются через специальные формы CMS.
2. Вся нужная информация постоянно в одном и том же месте. Пользователь может отправлять запрос на нужную информацию, а сайт мгновенно ее предоставлять с помощью AJAX запросов.
3. **Удобно использовать в больших проектах**. С помощью серверных скриптов можно воплотить практически любую требуемую функциональность.
4. **Легкое продвижение сайта**. На динамических ресурсах можно построить самые разные виды сайтов, в которых будут заинтересованы многие люди.

Недостатки динамических сайтов:

1. **Высокая стоимость создания и поддержки бекэнда (серверной части сайта)**.
2. **Повышенные требования к серверу**. Большая функциональность требует серьезных вычислительных мощностей для работы сайта без проблем с большим количеством людей.
3. **Дорогой хостинг**. Необходимы дополнительные услуги для управления базами данных, а также защита от взлома системы.

2 Генераторы статических сайтов

Генератор статических веб-сайтов – это программный инструмент, который превращает текстовые документы в набор связанных HTML-страниц. Принцип работы этого инструмента:

1. Берется какой-нибудь контент, например, новая запись в блоге.
2. Эти данные обрабатываются генератором и передаются в шаблон.
3. Сгенерированная статичная HTML-страница отправляется на хостинг.

Сейчас разработано много подобных генераторов, которые различаются по нескольким признакам:

1. Простота установки и настройки.
2. Наличие дополнительных расширений и плагинов.
3. Уровень поддержки разработчиком.
4. Способы развертывания сайта на хостинге.

Основная особенность во всех генераторах статических сайтов – это шаблоны, благодаря которым нет необходимости копировать один и тот же фрагмент HTML-кода в несколько разных файлов, все это автоматически делает генератор.

Кратко опишем несколько современных популярных генераторов статических сайтов [6].

2.1 Jekyll

Основная особенность данного генератора — это возможность держать все исходные файлы сайта в репозитории на сайте Github. Спустя всего несколько минут после публикации новых данных на Github они уже будут доступны для просмотра посетителям. Jekyll не имеет визуального редактора, чем может оттолкнуть начинающих веб-разработчиков. Однако это обдуманное решение, а не технический просчет. При оформлении записей

можно использовать языки Markdown и Textile, они хорошо читаемы и их несложно изучить.

Один из минусов статических сайтов – это отсутствие обратной связи. Однако это ограничение можно обойти, используя сторонние веб-сервисы. Например, для решения этой проблемы хорошо подойдет Wufoo. Этот сервис с простым и функциональным интерфейсом для создания форм и других дополнительных возможностей. На страницу добавляется JavaScript код, который подгружает данные с формой с сайта Wufoo. Имеется широкий выбор тарифов для разных нужд. Даже поиск можно реализовать через форму отправляющую запрос в Google с параметром `site`. Всех этих возможностей уже достаточно, чтобы создать неплохой блог с удобным перемещением.

Хостинг на Github имеет свои особенности. Git – это система управления версиями. Она спроектирована в виде набора программ, которая позволяет сохранять и отслеживать всю историю разработки проекта. Именно эта особенность является одной из наиболее весомых причин использовать Jekyll. Она дает возможность заменить административную панель CMS и совместить некоторые достоинства статических и динамических сайтов. Такой хостинг обеспечивает высокую скорость работы и простые операции загрузки, редактирования и удаления. В корневой папке репозитория можно разместить файл 404.html, который будет выдаваться пользователям, пытающимся открыть некорректную ссылку.

Для генерации веб-сайт нужно предварительно создать правила, на основе которых будет выполнена генерация. Некоторые из них уже записаны в самом Jekyll, например, в `_posts`, имена этих файлов обрабатываются правилами и соответствуют формату «дата-название». Это существенно облегчает разработку веб-сайта.

Можно писать собственные правила на языке шаблонов Liquid или же разрабатывать плагины на Ruby. Если размещать код на Github, то со вторым способом могут возникнуть проблемы, так как он запускает Jekyll в

безопасном режиме, блокирующем выполнение Ruby-скриптов. Но и для этого случая есть решение: можно сначала полностью сгенерировать сайт на своем персональном компьютере, а потом уже загрузить его на хостинг. Этот процесс можно автоматизировать с помощью shell-скрипта, который соберет сайт в папку с репозиторием, выполнит все изменения и отошлет их на сервер.

Ниже приведен пример простого сценария на Liquid, который выводит на страницу все посты (листинг 1).

Листинг 1. Вывод всех постов

```
{% for post in mysite.posts %}
<li><span>{{ post.date | date_to_string }}</span> &raquo; <a href="{{ post.url }}">{{ post.title }}</a></li>
{% endfor %}
```

Имеющийся в Jekyll функционал позволяет легко разработать сайт большим количеством статей. Поддержка такого проекта не требовательна ни к хостингу, ни к самому человеку, который будет обновлять информацию на веб-сайте.

В третьей главе данной работы приведен практический пример создания с помощью генератора Jekyll статического сайта (блога).

2.2 Другие популярные генераторы

По мимо Jekyll существует большое количество других генераторов статических сайтов. Далее будут разобраны их основные особенности.

Hyde задумывался как полный аналог Jekyll, однако написан он был на Python. Его название является отсылкой к популярному готическому роману Р.Л. Стивенсона «Странная история доктора Джекилла и мистера Хайда».

Существует две версии Hyde. Первая версия была основана на Django templates; в данный момент ее разработка приостановлена, последние обновления проекта в репозитории на GitHub были сделаны в 2009. Новая версия Hyde сейчас находится в активной разработке.

Функциональные возможности нового Hyde не отличаются от Jekyll, он так же поддерживает развертывание сайта на Github. В качестве недостатка можно отметить очень сжатую документацию к этому генератору небольшое число доступных плагинов и расширений. Hyde можно считать пока не доработанным проектом, хотя он имеет большой потенциал стать популярным в будущем.

Pelican, как и Hyde написан на Python. По сравнению с другими генераторами статических сайтов он имеет большой набор функций: работа с черновиками, интеграция с социальными сетями, добавление изображений, конвертация HTML-страниц в PDF, поддержка многоязычности и многое другое. Этот генератор очень хорошо подходит для ведения блогов. Посты можно писать в Markdown, а также в форматах reStructuredText и Asciidoc. Существует плагин для переноса всего сайта на Wordpress.

Pelican устанавливается с помощью менеджера пакетов pip. Поддерживается множество способов развертывания сайта: по FTP, по SSH, на Amazon S3, GitHub Pages, Dropbox и RackSpace Cloud Files.

Grow – довольно интересный и перспективный инструмент, написанный на Python. Чтобы установить Grow, необходимо всего лишь скачать скрипт с официального сайта; все пакеты будут установлены в автоматическом режиме.

В основе Grow лежит правило «конфигурация, а не код». Это означает, что для создания новый проекта (в терминологии Grow проекты называются подами, pods) необходимо скопировать на локальный компьютер тему, которая представляет собой репозиторий на GitHub. В этой теме находится набор конфигурационных файлов, с помощью которых описывается вся архитектура веб-сайта. При использовании такой темы программного кода писать не нужно. Настройки проекта хранятся в конфигурационном файле podspec.yaml. В нем указываются следующие параметры:

1. Метаданные проекта (имя и прочая информация).

2. Инструменты предварительной обработки (например, SASS, Closure Compliler или другие).
3. Данные по локализации сайта (в частности, список локалей, в которых сайт будет доступен).
4. Данные о статичных файлах и специальных страницах.
5. Настройки развертывания веб-сайта.

Работа с контентом осуществляется с помощью Markdown или HTML; хранится контент в директории /content. Структура страниц описывается в конфигурационных файлах в формате YAML, описание которого было выше. В процессе генерации сайта Grow создает страницы на основе написанных настроек конфигурации [8].

В Grow есть функция автоматического перевода текстовых фрагментов. Для этого используется библиотека Goslate library, работающая с Google Translate. Чтобы перевести сайт, достаточно просто выполнить команду `translate`.

Nanoblogger – этот генератор статических сайтов, ориентированный на создание блогов. Из прочих решений выделяется он тем, что написан на языке оболочки bash. Для создания статичных HTML-страниц в качестве основных инструментов этот генератор использует стандартные для Unix-систем утилиты командной строки `cat`, `grep` и `sed`.

Nanoblogger весьма прост, но по возможностям не уступаем другим генераторам, написанным на Python или Ruby. Этот генератор поддерживает Atom/RSS, с его помощью можно создать на сайте календарь, отсортировать посты по категориям, создать архив постов.

Устанавливается Nanoblogger максимально просто. Он уже включен в официальный репозитории большинства популярных дистрибутивов Linux и устанавливается при помощи стандартного менеджера пакетов.

С Nanoblogger удобно работать из командной строки. Он имеет подробную документацию, где описаны все команды. Синтаксис прост и

понятен. Исходный код также написан предельно просто и имеет хорошую гибкость. Всегда можно переписать код под конкретного пользователя и проекта. Для Nanoblogger существуют множество плагинов и расширений. Официальный набор плагинов (nanoblogger extras) также включен в официальные репозитории и устанавливается стандартным способом.

DocPad написан на CoffeeScript, поэтому для работы с ним на компьютере необходимо установить NodeJS.

Применяется этот генератор для блогов, но в отличие от других подобных продуктов он имеет более широкие возможности для применения. DocPad не является генератором статических сайтов в стандартном виде. Ему можно найти применение в роли разных инструментов: генератора, движка сайтов и шаблонизатора. Он оснащен довольно удобным интерфейсом, который позволяет использовать только те функции, которые нужны в данный момент; остальные всегда можно реализовать самостоятельно.

Неоспоримым преимуществом DocPad является очень подробная документация. Кроме того, на официальном сайте опубликованы заготовки, на основе которых пользователи могут создавать собственные сайты.

Для DocPad существует множество разнообразных плагинов. Из наиболее интересных расширений упомянуть WYSIWYG-редакторы и веб-интерфейсы, облегчающие публикацию постов в статическом блоге.

3 Пример создания сайта на Jekyll

Jekyll является одним из самых популярных генераторов статических сайтов, он имеет большое количество плагинов, расширений и шаблонов для самых разных тем. Именно по этим причинам я выбрал этот генератор для построения сайта [7].

3.1 Установка Jekyll

Чтобы работать в Jekyll третьей версии на операционной системе Windows нужно установить дополнительное программное обеспечение: интерпретатор языка Ruby версии 2.0 или выше, а также фреймворк RubyGems. Получить эти программы можно с официального сайта rubyinstaller.org. В операционных системах Mac OS и некоторых дистрибутивах Linux язык Ruby уже предустановлен или является частью системы.

Прежде чем перейти к установке Jekyll, следует проверить, какая версия Ruby была установлена, чтобы избежать возможных ошибок из-за несовпадений версий и разных сертификатов сервера. Сделать это можно с помощью команд `ruby -v` и `gem -v` (рисунок 1).

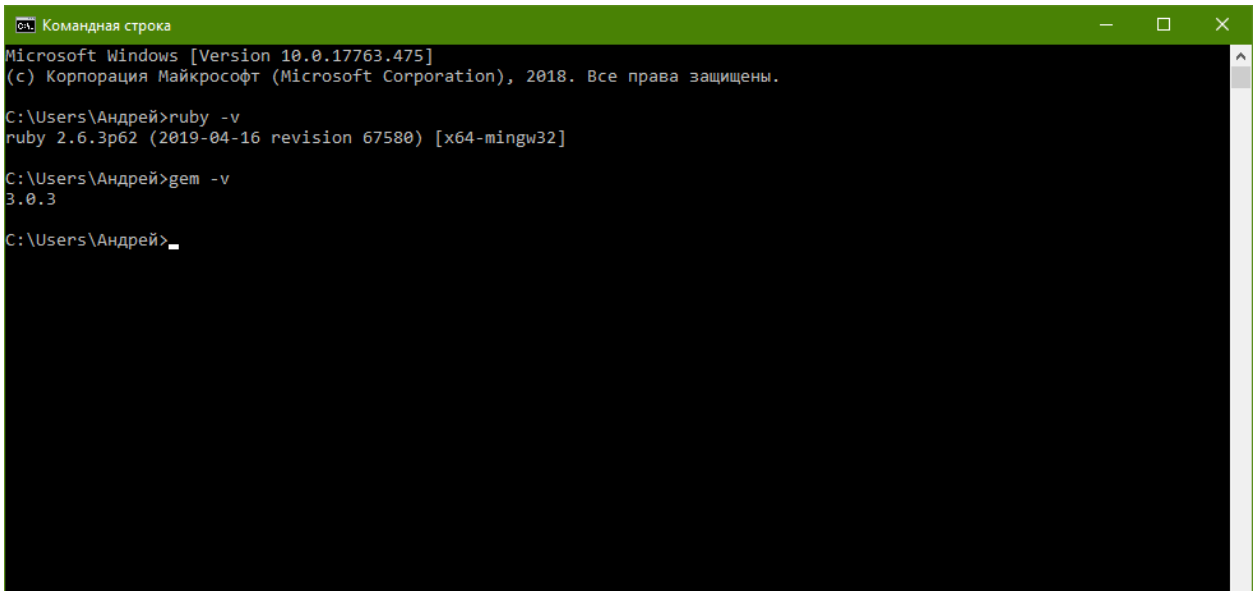


Рисунок 1 – Проверка версии Ruby и RubyGem

Далее можно перейти к установке самого генератора. Делается это очень просто, в командную строку нужно ввести команду `gem install Jekyll` (рисунок 2).

```
Командная строка
Installing ri documentation for sass-listen-4.0.0
Parsing documentation for sass-3.7.4
Installing ri documentation for sass-3.7.4
Parsing documentation for jekyll-sass-converter-1.5.2
Installing ri documentation for jekyll-sass-converter-1.5.2
Parsing documentation for concurrent-ruby-1.1.5
Installing ri documentation for concurrent-ruby-1.1.5
Parsing documentation for i18n-0.9.5
Installing ri documentation for i18n-0.9.5
Parsing documentation for http_parser.rb-0.6.0
unknown encoding name "chunked\r\n\r\n25" for ext/ruby_http_parser/vendor/http-parser-java/tools/parse_tests.rb, skippin
g
Installing ri documentation for http_parser.rb-0.6.0
Parsing documentation for eventmachine-1.2.7-x64-mingw32
Installing ri documentation for eventmachine-1.2.7-x64-mingw32
Parsing documentation for em-websocket-0.5.1
Installing ri documentation for em-websocket-0.5.1
Parsing documentation for colorator-1.1.0
Installing ri documentation for colorator-1.1.0
Parsing documentation for public_suffix-3.0.3
Installing ri documentation for public_suffix-3.0.3
Parsing documentation for addressable-2.6.0
Installing ri documentation for addressable-2.6.0
Parsing documentation for jekyll-3.8.5
Installing ri documentation for jekyll-3.8.5
Done installing documentation for safe_yaml, rouge, forwardable-extended, pathutil, mercenary, liquid, kramdown, ruby_de
p, ffi, rb-inotify, rb-fsevent, listen, jekyll-watch, sass-listen, sass, jekyll-sass-converter, concurrent-ruby, i18n, h
ttp_parser.rb, eventmachine, em-websocket, colorator, public_suffix, addressable, jekyll after 48 seconds
25 gems installed

C:\Users\Андрей>
```

Рисунок 2 – Установка Jekyll

После выполнения команды установилось большое количество файлов и пакетов (на рисунке 2 представлены не все установленные компоненты, они просто не умещаются на экране). Установка прошла удачно, теперь на персональном компьютере имеется готовый к работе Jekyll версии 3.8.5. Проверить версию можно командой `Jekyll -v` (рисунок 3).

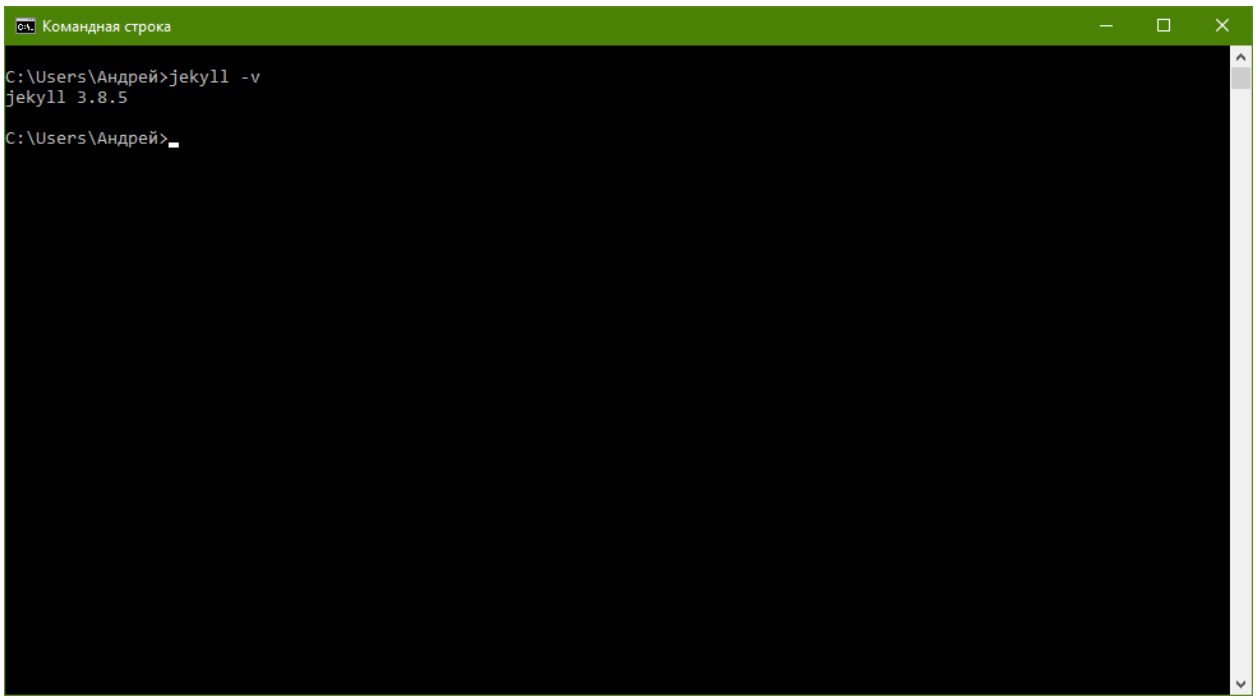


Рисунок 3 – Проверка версии Jekyll

3.2 Создание сайта

В качестве примера статического сайта мы создадим новостной блог (это позволит продемонстрировать основные возможности Jekyll).

Создание сайта начинается с выбора папки, в которой он будет храниться. Создадим эту папку из командной строки и перейдем в нее:

`cd /D D:\ВКР сайт.`

Следующий шагом будет создание нового проекта в выбранной директории с помощью команды `Jekyll new VKR_blog`. При этом вызывается специальная утилита Bundler, которая является частью Ruby. Она в автоматическом режиме устанавливает необходимые для сайта библиотеки (наборы классов) Ruby (рисунок 4), которые называются гемами (gems).

```
Командная строка
(c) Корпорация Майкрософт (Microsoft Corporation), 2018. Все права защищены.

C:\Users\Андрей>cd /D D:\ВКР сайт

D:\ВКР сайт>jekyll new VKR_blog
Running bundle install in D:/ВКР сайт/VKR_blog...
  Bundler: Fetching gem metadata from https://rubygems.org/..........
  Bundler: Fetching gem metadata from https://rubygems.org/.
  Bundler: Resolving dependencies...
  Bundler: Using public_suffix 3.0.3
  Bundler: Using addressable 2.6.0
  Bundler: Using bundler 1.17.2
  Bundler: Using colorator 1.1.0
  Bundler: Using concurrent-ruby 1.1.5
  Bundler: Using eventmachine 1.2.7 (x64-mingw32)
  Bundler: Using http_parser.rb 0.6.0
  Bundler: Using em-websocket 0.5.1
  Bundler: Using ffi 1.10.0 (x64-mingw32)
  Bundler: Using forwardable-extended 2.6.0
  Bundler: Using i18n 0.9.5
  Bundler: Using rb-fsevent 0.10.3
  Bundler: Using rb-inotify 0.10.0
  Bundler: Using sass-listen 4.0.0
  Bundler: Using sass 3.7.4
  Bundler: Using jekyll-sass-converter 1.5.2
  Bundler: Using ruby_dep 1.5.0
  Bundler: Using listen 3.1.5
  Bundler: Using jekyll-watch 2.2.1
  Bundler: Using kramdown 1.17.0
  Bundler: Using liquid 4.0.3
  Bundler: Using mercenary 0.3.6
  Bundler: Using pathutil 0.16.2
  Bundler: Using rouge 3.3.0
  Bundler: Using safe_yaml 1.0.5
  Bundler: Using jekyll 3.8.5
  Bundler: Fetching jekyll-feed 0.12.1
  Bundler: Installing jekyll-feed 0.12.1
  Bundler: Fetching jekyll-seo-tag 2.6.0
  Bundler: Installing jekyll-seo-tag 2.6.0
  Bundler: Fetching minima 2.5.0
  Bundler: Installing minima 2.5.0
  Bundler: Fetching tzinfo 2.0.0
  Bundler: Installing tzinfo 2.0.0
  Bundler: Fetching tzinfo-data 1.2019.1
  Bundler: Installing tzinfo-data 1.2019.1
  Bundler: Fetching wdm 0.1.1
  Bundler: Installing wdm 0.1.1 with native extensions
  Bundler: Bundle complete! 5 Gemfile dependencies, 32 gems now installed.
  Bundler: Use `bundle info [gemname]` to see where a bundled gem is installed.
New jekyll site installed in D:/ВКР сайт/VKR_blog.

D:\ВКР сайт>
```

Рисунок 4 – Выбор директории и создание исходных файлов сайта

В каталоге сайта появились конфигурационные файлы, но это все еще не полноценный сайт, а лишь набор правил, по которым он будет генерироваться. Для генерации сайта нужно ввести команду `Jekyll build`. Просмотреть созданный сайт можно с помощью сервера, запускаемого командой `Jekyll server` (рисунок 5).

```
Командная строка - jekyll server

D:\BKP сайт\VKR_blog>jekyll build
Configuration file: D:/BKP сайт/VKR_blog/_config.yml
            Source: D:/BKP сайт/VKR_blog
       Destination: D:/BKP сайт/VKR_blog/_site
 Incremental build: disabled. Enable with --incremental
      Generating...
       Jekyll Feed: Generating feed for posts
                    done in 0.494 seconds.
 Auto-regeneration: disabled. Use --watch to enable.

D:\BKP сайт\VKR_blog>jekyll server
Configuration file: D:/BKP сайт/VKR_blog/_config.yml
            Source: D:/BKP сайт/VKR_blog
       Destination: D:/BKP сайт/VKR_blog/_site
 Incremental build: disabled. Enable with --incremental
      Generating...
       Jekyll Feed: Generating feed for posts
                    done in 0.502 seconds.
 Auto-regeneration: enabled for 'D:/BKP сайт/VKR_blog'
    Server address: http://127.0.0.1:4000/
  Server running... press ctrl-c to stop.
```

Рисунок 5 – Генерация сайта и запуск сервера

Созданный стандартный сайт Jekyll располагается по локальному адресу http://127.0.0.1:4000/. При обращении к этому адресу мы увидим одну запись с приветствием посетителя (рисунок 6).

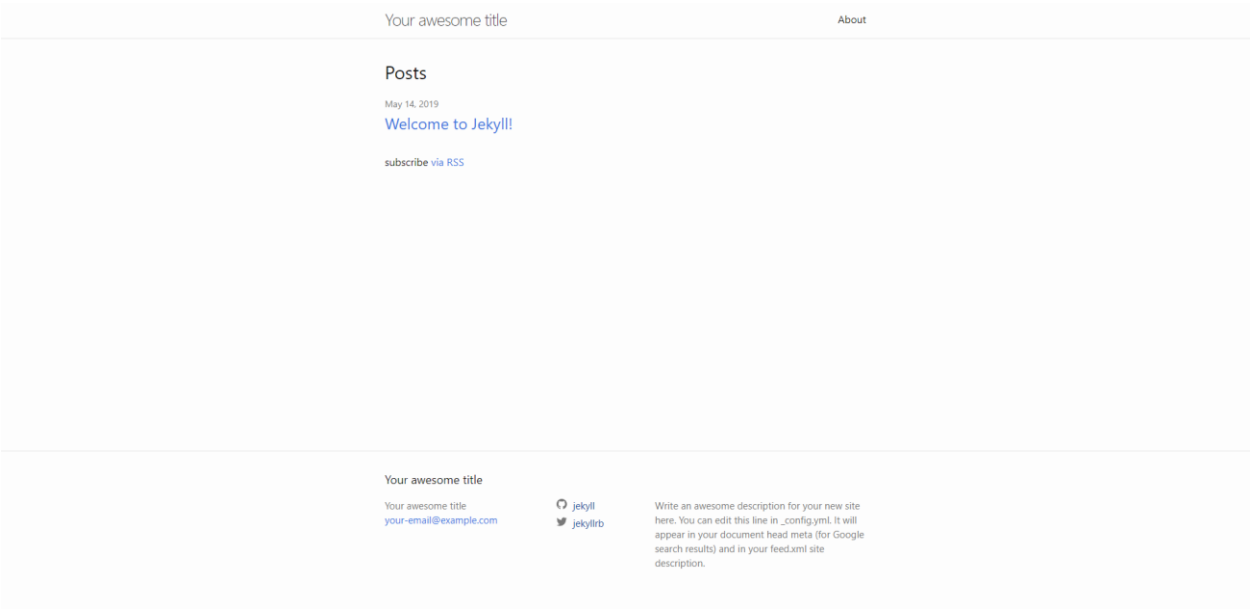


Рисунок 6 – Сгенерированный сайт

Для генерации сайта Jekyll использует Liquid – язык шаблонов. Простыми словами его можно описать как мост между данными сайта и HTML-шаблонами. Пользуясь простыми конструкциями, с помощью Liquid можно получить доступ к данным сайта (к заголовку статьи, картинок и тегов) и выводить эти данные в шаблон. Предварительно записав нужную информацию в переменную, больше не нужно волноваться о том правильно ли отображается информация на экране. Это одно из главных преимуществ этого языка. После того как система обработает шаблон для вывода, происходит анализ шаблона и CSS на предмет поиска заглушек Liquid. После того как нужные элементы найдены, система заменяет заглушки на необходимую информацию из заполненных раннее переменных. Заглушка представляет собой фрагмент кода, который изменяется после обработки браузером. Существует два вида заглушек: первый – двойные фигурные скобки `{{ }}` отображают вывод и второй – фигурные скобки плюс знак процента `{% %}` определяет логические конструкции. Liquid дает возможность контролировать что именно будет выведено на экран в зависимости от ситуации, например, если товар отсутствует на складе, то с помощью логических условий он не будет отображаться в каталоге [15].

Для создания функционального новостного блога нужен специальный шаблон, отвечающий основным требованиям блога. Остальные дополнительные возможности вроде разметки, стиля и добавления комментариев можно будет дописать самому. Jekyll – генератор, ориентированный на блоги, поэтому проблем с выбором шаблона не возникнет. Установить шаблон очень просто, для этого необходимо распаковать скачанные файлы в каталог сайта. Все исходные данные можно изменять, подгоняя дизайн сайта под самые разные нужды, в этом примере были изменены размер постов, картинок, количество постов на странице, а также добавлены дополнительные функции (глава 3.4).

Эти файлы включают в себя настройки конфигурации и уже созданный набор html-страниц. Главным файлом конфигурации является _config.yml. Он содержит в себе следующую информацию (листинг 2):

- используемые библиотеки: `plugins`
- название сайта: `title`
- ссылки: `links`
- количество статей на странице: `paginate`
- предустановки `markdown`
- ключевые слова, по которым поисковая система будет выдавать результат: `keywords`
- описание сайта: `description`

Листинг 2. Файл _config.yml

```
# Site settings
## Change the values as needed.
title: Новостной блог

description: "A jekyll theme implementing material design by using muicss."
keywords: "jekyll, theme, material, muicss blog"
url: "http://bitwiser.in/bitwiser-material" # the base hostname & protocol for your site
plugins: [jekyll-paginate]
baseurl: "" # the subpath of your site, e.g. /blog/
paginate: 3
paginate_path: "page:num"
# Build settings
markdown: kramdown
kramdown:
  input: GFM
  syntax_highlighter: rouge
sass:
  sass_dir: _sass
  style: :compressed
links:
  vk:id146341441
exclude:
  - bower_components
  - README.md
```

Все шаблонные страницы хранятся в папках _layouts и _includes. На основе этих файлов генерируются записи блога. Самым часто используемым шаблоном будет posts.html, далее будет подробно разобрана его структура (листинг 3).

Листинг 3. Пример шаблона постов

```
layout: base
<body style="padding: 0">
     <article class="post-full">
    <header class="bg-holder post-page {% if page.color %}bg-{{
page.color }}{% endif %}" style="{% if page.cover
%}background:url({{page.cover}}) no-repeat center center;
background-size: cover{% endif %}">
          <a href="{{site.url}}/" class="overlay"><i class="fa
fa-home fa-2x"></i></a>
          <div class="img-overlay"></div>
          <!-- Put this script tag to the <head> of your page -->
    </header>
    <div class="mui-container mui-panel post-content">
          <h1 class="article-title"><a href="{{page.url}}">{{
page.title }}</a></h1>
          <p class="post-meta"> <a href="{{ site.url }}">
          </a><time datetime="{{ page.date | date: "%Y-%m-
%dT%H:%M" }}"><i class="fa fa-calendar"></i>{{ page.date | date:
"%d.%m.%Y" }}</time>
          </p>
        <p class="post-tag-container">
            {% for tag in page.tags %}
            <span class="post-tag"><i class="fa fa-tag"></i>{{
tag }}</span>
            {% endfor %}
        </p>
      <div class="post-html">
          {{ content }}
      </div>
      </div>
  </article>
```

Именно этот файл будет обрабатывать статьи, с помощью него страницы будут иметь единый формат. На данной странице указываются все правила, по которым будут отображаться статьи. Сюда подгружаются все данные о статье:

- Название: `page.title`
- Теги: `tag`
- Миниатюра страницы: `site.url`

- Стили: `style`
- Текст: `content`

Все это обрабатывается и выводится в указанные места.

Все статьи хранятся в отдельной папке _posts в формате markdown. Далее будет подробно разобран процесс создания статьи (листинг 4).

Листинг 4. Пример статьи

```
---
layout: post
color: deep-purple
cover: "https://www.stevsky.ru/dibujos/2019/04/notre-dam_2.jpg"
title:  "Сгорел собор парижской богоматери"
date:   2019-04-16 13:50:39
tags: Пожар Париж Трагедия
categories: jekyll update
---
(текст статьи)
```

Статья должна содержать следующие данные:

- Название шаблона, по которому она будет генерироваться: `layout`.
- Цвет миниатюры (если нет картинки): `color`.
- Картинка для миниатюры: `cover`.
- Заголовок статьи: `title`.
- Тэги: `tags`.
- Категории: `categories`.

Так как страницы шаблонов – это обычные HTML-страницы, то для изменения на сайте стиля, оформления, шрифтов и прочего можно использовать CSS [9].

Также Jekyll имеет поддержку простого языка разметки Markdown для написания статей. Этот язык позволяет определять в тексте:

- Заголовки разного уровня.
- Списки со вложениями.
- Цитаты.
- Исходного код на языке программирования.
- Ссылки.
- Курсивный, жирный и зачеркнутый текст.
- Изображения.

Таким образом, после создания всех этих файлов можно сгенерировать сайт. В результате создается новая папка _site, в которой хранятся все сгенерированные исходные файлы. Запустив локальный сервер, можно посмотреть на результат проделанной работы (рисунок 7).

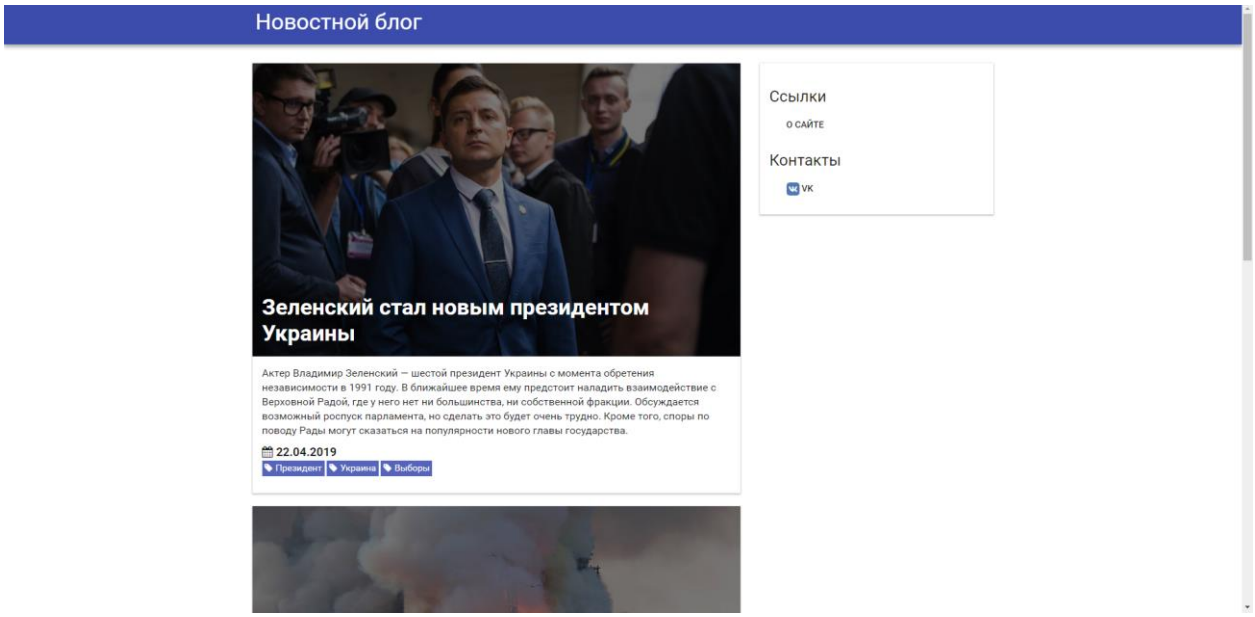
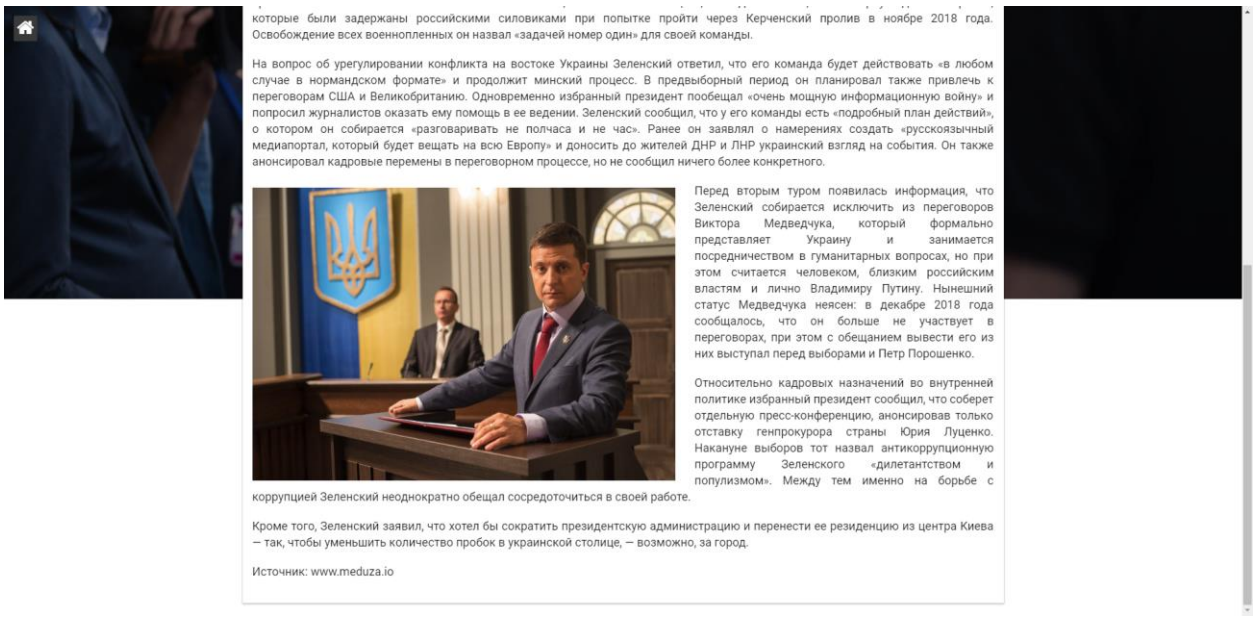


Рисунок 7 – Домашняя страница

Каждую статью можно прочитать статью в расширенном виде, щелкнув мышью по ее заголовку (рисунок 8).



которые были задержаны российскими силовиками при попытке пройти через Керченский пролив в ноябре 2018 года. Освобождение всех военнопленных он назвал «задачей номер один» для своей команды.

На вопрос об урегулировании конфликта на востоке Украины Зеленский ответил, что его команда будет действовать «в любом случае в нормандском формате» и продолжит минский процесс. В предвыборный период он планировал также привлечь к переговорам США и Великобританию. Одновременно избранный президент пообещал «очень мощную информационную войну» и попросил журналистов оказать ему помощь в ее ведении. Зеленский сообщил, что у его команды есть «подробный план действий», о котором он собирается «разговаривать не полчаса и не час». Ранее он заявлял о намерениях создать «русскоязычный медиапортал, который будет вещать на всю Европу» и доносить до жителей ДНР и ЛНР украинский взгляд на события. Он также анонсировал кадровые перемены в переговорном процессе, но не сообщил ничего более конкретного.

Перед вторым туром появилась информация, что Зеленский собирается исключить из переговоров Виктора Медведчука, который формально представляет Украину и занимается посредничеством в гуманитарных вопросах, но при этом считается человеком, близким российским властям и лично Владимиру Путину. Нынешний статус Медведчука неясен: в декабре 2018 года сообщалось, что он больше не участвует в переговорах, при этом с обещанием вывести его из них выступал перед выборами и Петр Порошенко.

Относительно кадровых назначений во внутренней политике избранный президент сообщил, что соберет отдельную пресс-конференцию, анонсировав только отставку генпрокурора страны Юрия Луценко. Накануне выборов тот назвал антикоррупционную программу Зеленского «дилетантством и популизмом». Между тем именно на борьбе с коррупцией Зеленский неоднократно обещал сосредоточиться в своей работе.

Кроме того, Зеленский заявил, что хотел бы сократить президентскую администрацию и перенести ее резиденцию из центра Киева — так, чтобы уменьшить количество пробок в украинской столице, — возможно, за город.

Источник: www.meduza.io

Рисунок 8 – Статья

Для того, чтобы сайт был более привлекательным, нужно создать отдельную страницу для ошибки 404. Эта ошибка возникает при неудачной попытке открыть страницу сайта (пользователь ввел неправильный адрес или переходит по несуществующей ссылке). Можно обойтись и без дополнительной страницы, браузер в этом случае сам сгенерирует ее. Но такая страница будет невзрачной и появляется большая вероятность что пользователь просто уйдет с сайта. Для предотвращения возможных потерь пользователя лучше создать что-то свое, чтобы пользователь не был в замешательстве и быстро вернулся на сайт (рисунок 9).

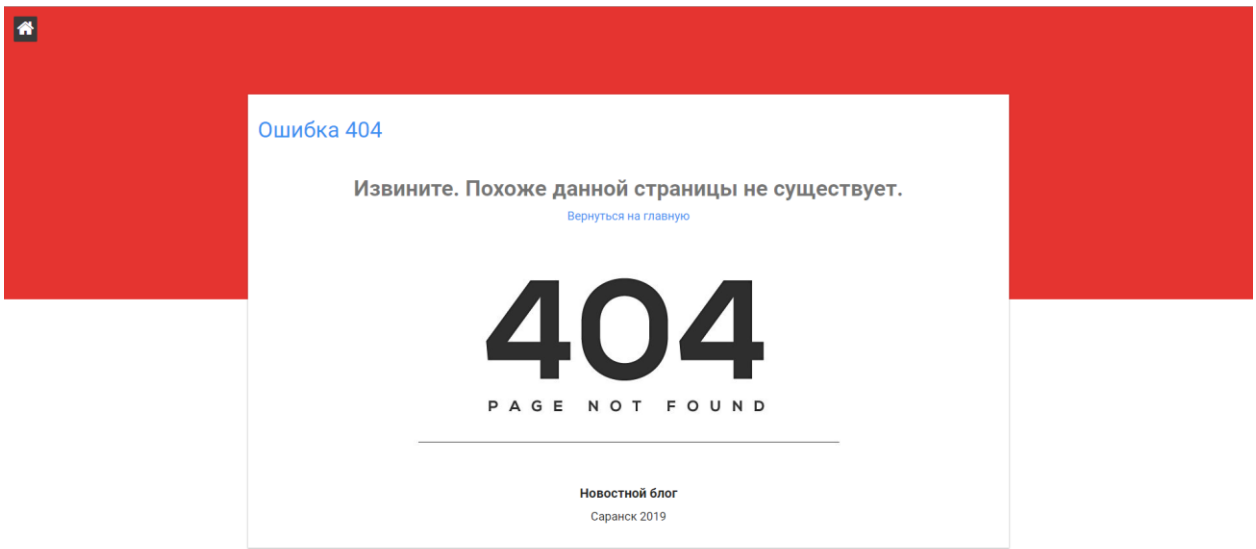


Рисунок 9 – Ошибка 404

Данная страница по своей структуре является статьей, она будет вызываться, когда пользователь перейдет по несуществующей ссылке.

3.3 Развертывание сайта на GitHub

Чтобы новостной блог могли просматривать другие пользователи, его опубликовать в Интернете. Как уже говорилось ранее, сайт на Jekyll можно развернуть в виде репозитория на GitHub. При этом варианте развертывания сайта за хостинг не нужно платить деньги, при этом мы имеем функцию авторегенерации сайта: при редактировании содержимого сайта или добавлении новой статьи пользователи через пару минут увидят изменения в Интернете на сайте [12].

Сначала на локальный компьютер нужно установить систему управления версиями Git. Скачать его можно с официального сайта https://git-scm.com/, установка осуществляется стандартным образом [13].

После установки необходимо убедиться, что установлена последняя версия Git. Для этого в командной строке нужно ввести команду `git --version` (рисунок 10).

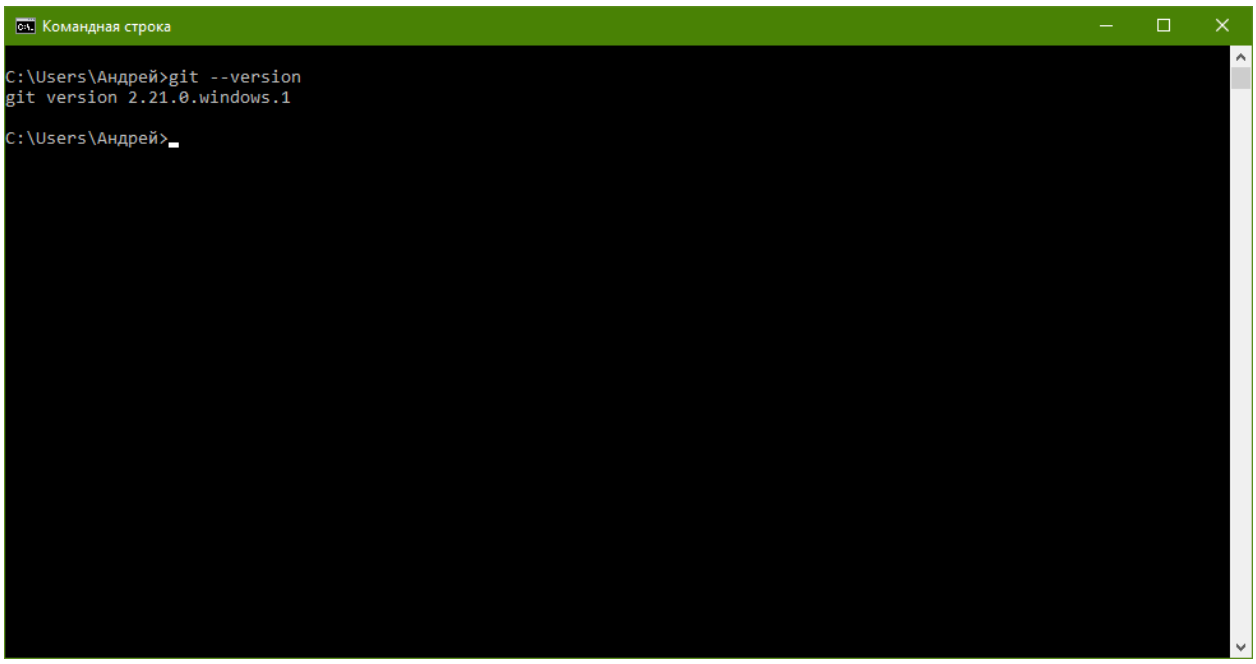


Рисунок 10 – Проверка версии Git

Далее нужно зарегистрировать аккаунт на GitHub и создать там новый репозиторий, имя которого будет начинаться с созданного аккаунта. В нашем случае репозиторий будет называться default1997.github.io. Содержимое этого репозитория будет доступно всем пользователям Интернета по адресу https://default1997.github.io.

Чтобы соединить локальный репозиторий и репозиторий GitHub, нужно, находясь в папке, где хранится сайт, выполнить команду `git clone` и в параметре команды указать ссылку на репозиторий GitHub (рисунок 11). В результате создастся новая папка с таким же названием, как и репозиторий на GitHub: default1997.github.io [14].

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.17763.475]
(c) Корпорация Майкрософт (Microsoft Corporation), 2018. Все права защищены.

D:\ВКР сайт>git clone https://github.com/Default1997/default1997.github.io.git
Cloning into 'default1997.github.io'...
warning: You appear to have cloned an empty repository.

D:\ВКР сайт>
```

Рисунок 11 – Создание репозитория GitHub

Эта папка пуста, в нее нужно скопировать исходные файлы сайта. Следующим шагом будет индексирование всех файлов для добавления в репозиторий. Для этого в командной строке нужно выполнить команду `git add` и сразу проверить статус файлов командой `git status` (рисунок 12). Файлы должны показываться как вновь созданные.

```
C:\Windows\System32\cmd.exe
Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   .gitignore
        new file:   0.1.0'
        new file:   404.html
        new file:   LICENSE
        new file:   README.md
        new file:   _config.yml
        new file:   _includes/analytics.html
        new file:   _includes/footer.html
        new file:   _includes/head.html
        new file:   _includes/header.html
        new file:   _includes/sidebar.html
        new file:   _layouts/base.html
        new file:   _layouts/default.html
        new file:   _layouts/page.html
        new file:   _layouts/post.html
        new file:   _posts/2015-07-12-color-choices.md
        new file:   _posts/2015-07-12-demo-post.md
        new file:   _posts/2019-16-04-paris.md
        new file:   _posts/2019-22-04-zelensky.md
        new file:   _sass/_colors.scss
        new file:   _sass/_mui.scss
        new file:   _sass/_syntax-highlighting.scss
        new file:   about.md
        new file:   css/main.scss
        new file:   demo.md
        new file:   feed.xml
        new file:   img/sharer.png
```

Рисунок 12 – Выбор и проверка исходных файлов

Далее нужно сделать зафиксировать файлы в репозитории с помощью команды `git commit -m “1.0”`.

Остался последний этап, все подготовленный файлы следует выгрузить в удаленный репозиторий на GitHub командой `git push -u origin master` (рисунок 13).

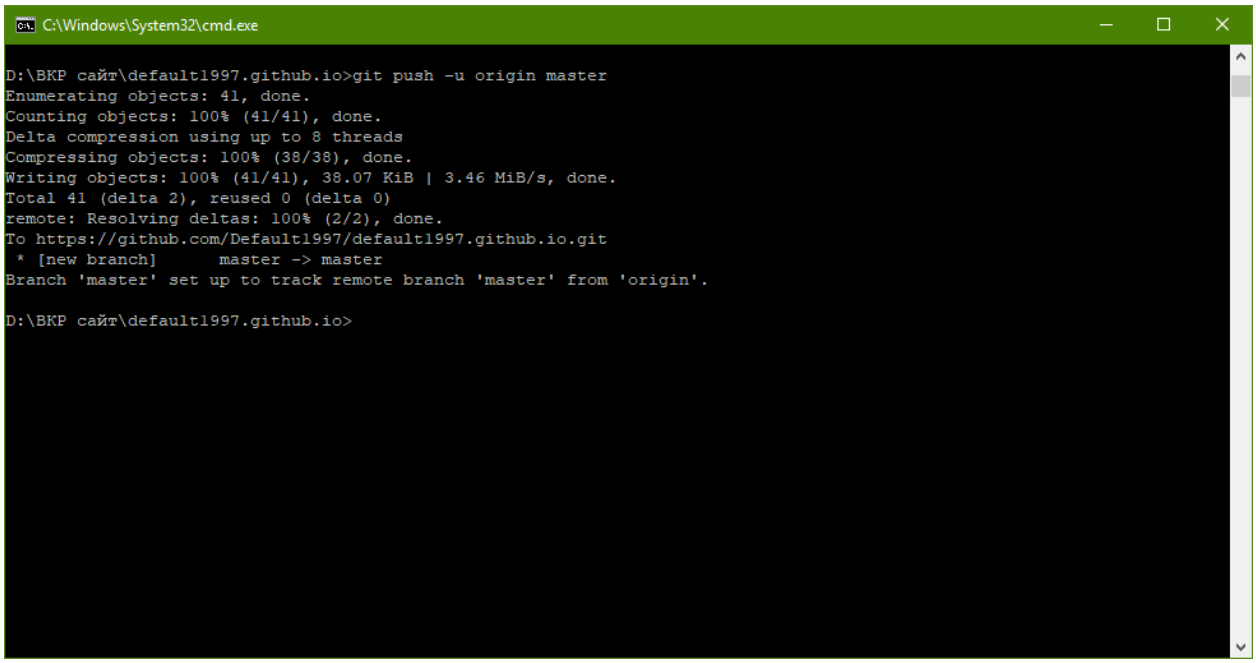


Рисунок 13 – Отправление исходных файлов в репозиторий GitHub

Теперь весь исходный код сайта располагается в репозитории GitHub, для корректной работы нужно лишь подправить адреса к CSS файлам. Сайт с блогом доступен в Интернете по стандартном адресу https://default1997.github.io.

Хостинг на GitHub бесплатный, но стандартное имя домена третьего уровня не очень подходит для продвижения сайта, поэтому лучше использовать свой домен. Получить персональное доменное имя второго уровня можно тоже бесплатно. Мы для своего сайта выбрали доступный домен vkrblog.ml, прошли регистрацию и настроили DNS сервер (рисунок 14).

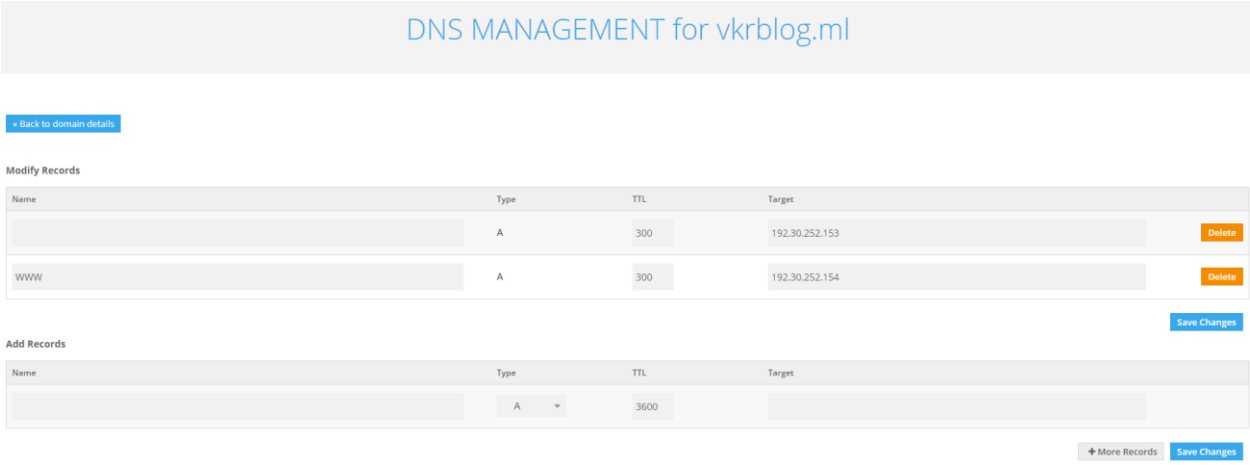


Рисунок 14 – Настройка DNS сервера

После регистрации персонального домена нужно указать его в настройках репозитория GitHub. Теперь наш сайт имеет более привлекательный адрес https://vkrblog.ml.

3.4 Дополнительные функции

Для успешного продвижения новостного блога в Интернете на сайте должны быть реализованы дополнительные функции: возможность оставить комментарий к посту в блоге или поделиться понравившейся информацией в социальных сетях. Для этого на странице статьи можно создать специальную кнопку и дописать код на языке JavaScript (листинг 5) [11].

Листинг 5. Скрипт и кнопка «Поделиться»

```
<script type="text/javascript"
src="https://vk.com/js/api/share.js?95" charset="windows-
1251"></script>
----------------------
<script type="text/javascript">
document.write(VK.Share.button(false,{type: "custom", text:
"<img src=\"https://vk.com/images/share_32.png\" width=\"32\"
height=\"32\" />"}));</script>
```

Социальная сеть Вконтакте поддерживает возможность «Поделиться» по умолчанию, поэтому достаточно написать путь к главному скрипту в тег <head> и вставить вторую часть кода в то место HTML-разметки, где должна

располагаться кнопка. В результате проделанных действий на сайте появляется возможность поделиться информацией с социальной сетью (Рисунок 15).

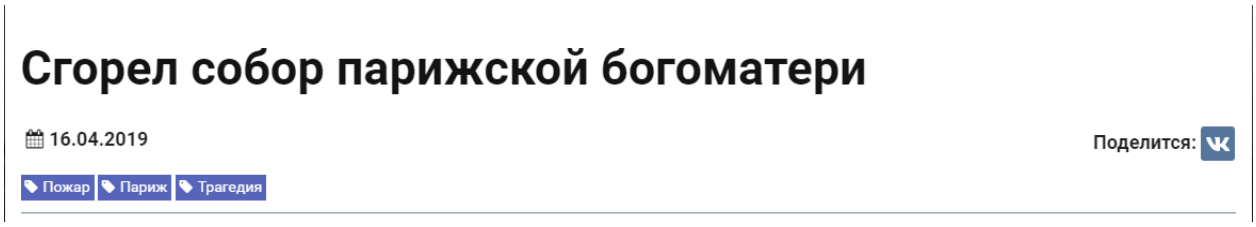


Рисунок 15 – Кнопка «Поделиться»

После щелчка по кнопке появится новое окно, при этом пользователю не нужно авторизоваться во ВКонтакте, если он сделал это ранее в том же браузере, ему остается лишь получателей сообщения и написать комментарий (рисунок 16).

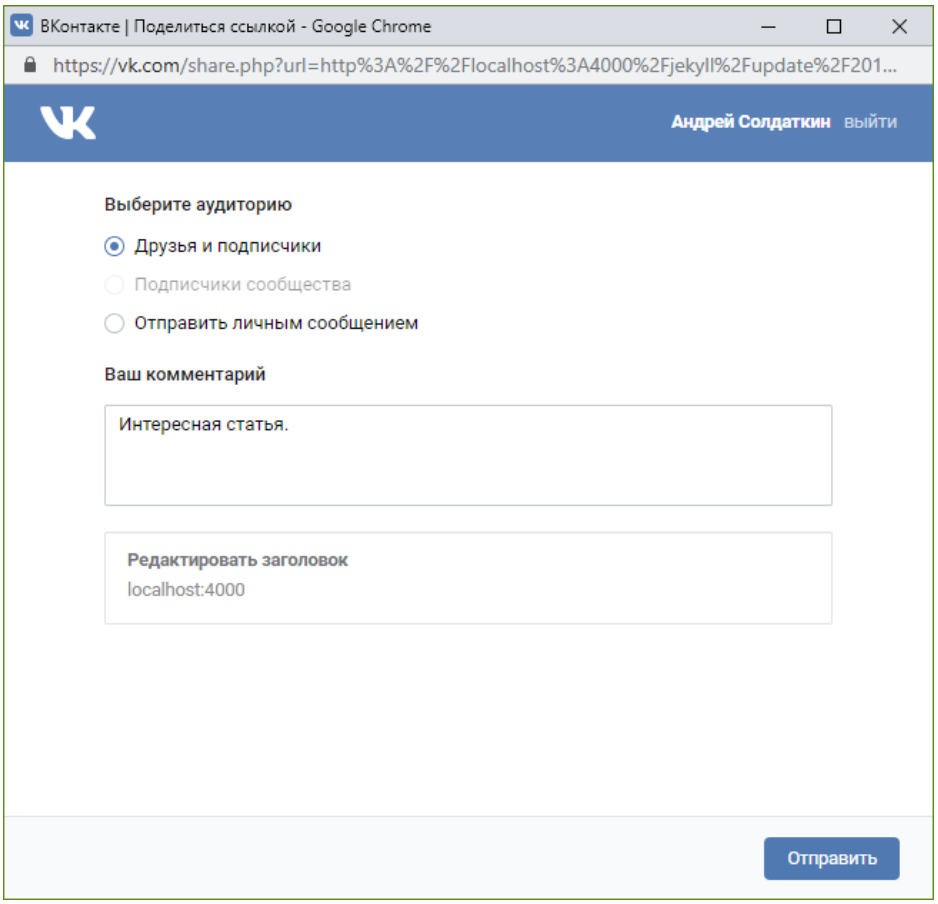


Рисунок 16 – Окно «Поделиться»

Чтобы статьи в новостном блоге становились популярнее, авторам необходимо иметь обратную связь с читателями. Статические сайты не имеют базы данных и поэтому такие возможности, как личный кабинет или комментирование статей, на них недоступны.

Однако можно воспользоваться сторонними сервисами, например, социальная сеть ВКонтакте бесплатно предоставляет виджет для комментариев. Для реализации этой функции на своем сайте необходимо получить специальный идентификатор API ID. ВКонтакте выдают его после того, как будет указан адрес, название и направленность сайта. Затем JavaScript-код со ссылкой на виджет нужно вставить на свою страницу (листинг 6).

Листинг 6. Скрипты для создания комментариев

```
<script type="text/javascript"
src="https://vk.com/js/api/openapi.js?160"></script>
---

<script type="text/javascript">
  VK.init({apiId: 6987350, onlyWidgets: true});
</script>
---
<div id="vk_comments"></div>
<script type="text/javascript">
VK.Widgets.Comments("vk_comments", {limit: 10, attach: "*"});
</script>
```

Как и в случае с кнопкой «Поделиться», в этом коде сначала нужно указать адрес к главному скрипту openapi.js, который располагается на серверах Вконтакте (первая часть кода должна располагаться в теге <head>). Вторая часть кода определяет идентификатор, он нужен для создания связи между сайтом и аккаунтом администратора во Вконтакте (этот код должен располагаться в теге <body>). Третью часть кода нужно вставить туда, где должны располагаться комментарии на сайте (рисунок 17).

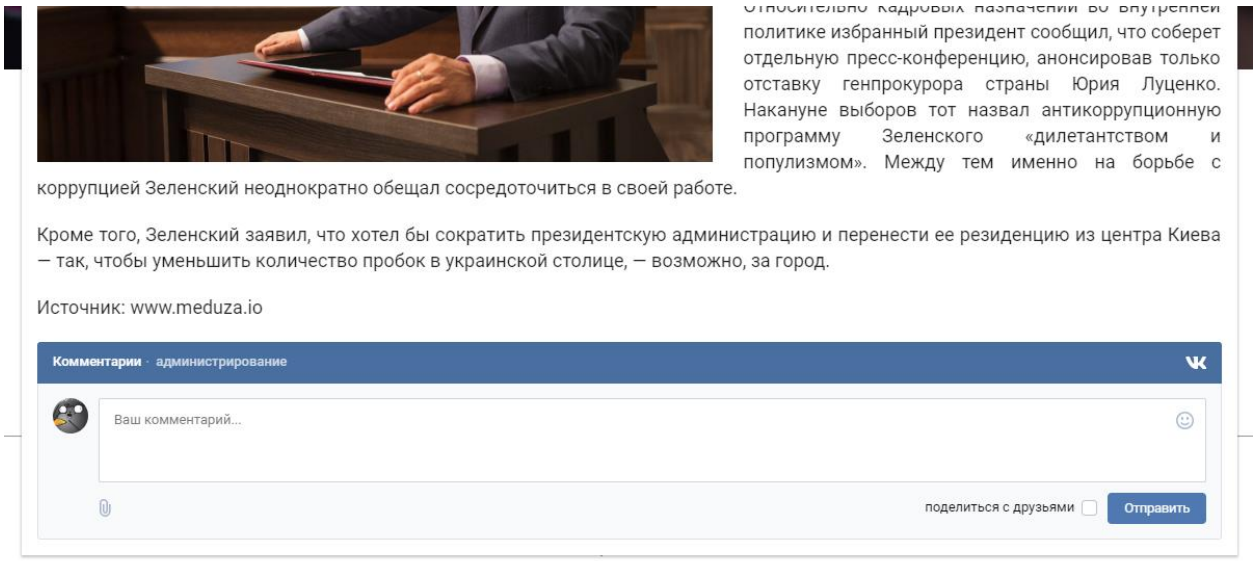


Рисунок 17 – Комментарии

Теперь у пользователей есть возможность комментирования с большим функционалом от ВКонтаке. Они могут прикреплять фотографии, видео, аудиозаписи к своему сообщению, а также если они захотят, то они могут поделится своим комментарием с друзьями, что положительно сказывается на популяризации сайта. Это очень удобно, ведь пользователям не нужно регистрироваться на сайте, тратить свое время, запоминать еще один логин и пароль, достаточно лишь быть авторизованным в социальной сети Вконтакте.

ЗАКЛЮЧЕНИЕ

С каждым годом появляются новые языки программирования, фреймворки и технологии для разработки и публикации сайтов в сети Интернет. Количество информации в глобальной сети очень быстро растет и одной из главных проблем становится обеспечение доступности веб-ресурсов в условиях высокой нагрузки на сайты.

Один из путей решения этой проблемы – использование автоматически генерируемых статических сайтов. На вход такой генератор получает текстовую информацию, а на выходе получается набор связанных HTML-страниц, которые размещаются на хостинге. В этом случае нагрузка на веб-сервер будет минимальной, страницы загружаются очень быстро. Кроме того, такие сайты имеют высокий уровень безопасности, нет риска того, что злоумышленники проведут DDOS-атаки на сервер. Конечно, функционал статических сайтов сильно ограничен, однако их возможности можно наращивать с помощью сторонних сервисов.

В рамках проделанной работы был создан статический сайт для новостного блога, в качестве генератора был выбран Jekyll. Были спроектированы шаблоны для генерации страниц сайта, а также реализованы функции для работы с отдельными статьями. В качестве дополнения к функционалу статического сайта были подключены некоторые сервисы социальной сети Вконтакте (комментарии для обратной связи и возможность поделиться ссылкой на понравившуюся статью).

Авторы генераторов статических сайтов прилагают максимум усилий, чтобы у разработчиков был широкий набор инструментов для построения функционального и привлекательного сайта. Для оформления внешнего вида используются HTML и CSS, контент сайта создается с помощью языка разметки (например, Markdown) и какого-либо шаблонизатора (например, Liquid). Синтаксис и принцип работы этих языков и инструментов прост и понятен, любой начинающий разработчик сможет быстро с ними разобраться.

Поэтому автоматически генерируемые статические сайты являются хорошим, несложным и надежным решением для создания небольших информационных сайтов и блогов.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Дунаев В. HTML, скрипты и стили / В. Дунаев. – СПб.: Вильямс, 2006. – 811 с.

2. Круг С. Веб-Дизайн: книга Стива Круга или "не заставляйте меня думать!"/ С. Круг – М.: Символ-Плюс, 2008. – 224 с.

3. Бабаев А. Создание сайтов / А. Бабаев , Б. Михаил, Е. Николай. – М.: Питер, 2014. – 410 с.

4. Нильсен Я. Web-дизайн. Удобство использования Web-сайтов / Нильсен Я., Х. Лоранжер. – СП.: Вильямс, 2009. – 376 с.

5. Феличи Д. Типографика. Шрифт, верстка, дизайн / Д. Феличи. – СПб.: Вильямс, 2018. – 496 с.

6. Выбираем генератор статических сайтов– [Электронный ресурс]: ресурс о программировании. – Режим доступа: https://habr.com/ru/company/selectel/blog/236441/.

7. Практическое руководство по Jekyll – [Электронный ресурс]: ресурс о программировании. – Режим доступа: https://habr.com/ru/post/207650/.

8. Практическое руководство по Markdown – [Электронный ресурс]: ресурс о программировании. – Режим доступа: https://paulradzkov.com/2014/markdown_cheatsheet/.

9. Справочник по CSS – [Электронный ресурс]: рес урс о программировании. – Режим доступа: http://htmlbook.ru/CSS.

10. Статические и динамические сайты – [Электронный ресурс]: ресурс о программировании. – Режим доступа: https://barabyn.ru/blog/kompyutery-i-internet/staticheskie-i-dinamicheskie-sajty-v-chem-raznica.html.

11. Виджеты Вконтакте – [Электронный ресурс]: ресурс о программировании. – Режим доступа: https://vk.com/dev/sites.

12. GitHub как хостинг – [Электронный ресурс]: ресурс о программировании. – Режим доступа: https://htmlacademy.ru/blog/99-github-as-hosting.

13. Git: руководство для начинающих – [Электронный ресурс]: ресурс о программировании. – Режим доступа: https://proglib.io/p/git-for-half-an-hour/.

14. Как начать работать с GitHub: быстрый старт

– [Электронный ресурс]: ресурс о программировании. – Режим доступа: https://habr.com/ru/post/125799/.

15. Знакомство с Liquid – [Электронный ресурс]: ресурс о программировании. – Режим доступа: https://www.internet-technologies.ru/articles/znakomstvo-s-liquid-yazykom-shablonov-shopify.html.

ПРИЛОЖЕНИЕ А
(обязательное)
Основные части кода веб-сайта

Конфигурация Jekyll

```
# Site settings
## Change the values as needed.
title: Новостной блог

description: "A jekyll theme implementing material
design by using muicss."
keywords: "jekyll, theme, material, muicss"
url: "" # the base hostname & protocol for your site
plugins: [jekyll-paginate]
baseurl: "" # the subpath of your site, e.g. /blog/
paginate: 3
paginate_path: "page:num"
# Build settings
markdown: kramdown
kramdown:
  input: GFM
  syntax_highlighter: rouge
sass:
  sass_dir: _sass
  style: :compressed
links:
  vk:id146341441
exclude:
  - bower_components
  - README.md
full_name: Brijesh
```

Домашняя страница

```
---
layout: default
---
<div class="home mui-row">
  <div class="mui-col-md-8">
  {% for post in paginator.posts %}
    <article class="mui-panel">
      <div class="bg-holder {% if post.color %}bg-{{
post.color }}{% endif %}"
```

```
style="background:url({{post.cover}}); background-size:
cover;">
    <h2 class="post-list-title">
 <a class="post-link" href="{{ post.url | prepend:
site.baseurl }}">{{ post.title }}</a>
        </h2>
        {% if post.cover %}<div class="img-
overlay"></div>{% endif %}
        <a href="{{ post.url | prepend: site.url }}"
class="overlay" ></a>
      </div>
      <div class="post-data">
        <p class="post-excerpt">{{ post.excerpt |
strip_html }}</p>
        <div class="post-meta">
            <time datetime="{{ post.date | date: "%Y-
%m-%dT%H:%M" }}"><i class="fa fa-calendar"></i>{{
post.date | date: "%d.%m.%Y" }}</time>
        </div>
        {% if post.tags and post.tags.size > 0 %}
        <p>
            {% for tag in post.tags %}
            <span class="post-tag"><i class="fa fa-
tag"></i>{{ tag }}</span>
            {% endfor %}
        </p>
        {% endif %}
      </div>
    </article>
  {% endfor %}
  <div class="paginator">
    {% if paginator.previous_page %}
    <a href="{{ paginator.previous_page_path }}"
class="mui-btn mui-btn-primary"><i class="fa fa-
chevron-circle-left"></i>Previous</a>
    {% endif %}
    <span>Page: {{ paginator.page }} of {{
paginator.total_pages }}</span>
    {% if paginator.next_page %}
    <a href="{{ paginator.next_page_path }}"
class="mui-btn mui-btn-primary">Next<i class="fa fa-
chevron-circle-right"></i></a>
    {% endif %}
```

```
 </div>
  </div>
{% include sidebar.html %}</div>
```

Общий HEAD

```
<head>
  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width,
initial-scale=1">
  <title>{% if page.title %}{{ page.title }} - {{
site.title }}{% else %}{{ site.title }}{% endif
%}</title>
  <meta name="description" content="{% if page.excerpt
%}{{ page.excerpt | strip_html | strip_newlines |
truncate: 160 }}{% else %}{{ site.description |
strip_newlines }}{% endif %}">

  <link
href="//fonts.googleapis.com/css?family=Roboto:300,400,
400italic,500,700" rel="stylesheet" type="text/css" />
  <link href="//cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.2.0/css/font-awesome.min.css"
rel="stylesheet" type="text/css" />

  <link rel="stylesheet" href="{{ "/css/main.css" |
prepend: site.baseurl | prepend: site.url }}">
  <link rel="canonical" href="{{ page.url |
replace:'index.html','' | prepend: site.baseurl |
prepend: site.url }}">
  <link rel="alternate" type="application/rss+xml"
title="{{ site.title }}" href="{{ "/feed.xml" |
prepend: site.baseurl | prepend: site.url }}" />
  <script src="{{ "/js/mui.min.js" | prepend:
site.baseurl | prepend: site.url }}"></script>
  <style>
      body {
        font-family: "Roboto", "Helvetica Neue",
Helvetica, Arial;
      }
  </style>
```

```
<meta property="og:title" content="{% if page.title
%}{{ page.title }} - {{ site.title }}{% else %}{{
site.title }}{% endif %}" />
  <meta name="twitter:title" content="{% if page.title
%}{{ page.title }} - {{ site.title }}{% else %}{{
site.title }}{% endif %}" />

{% if page.tags and page.tags.size > 0 %}
  <meta name="keywords" content="{% for tag in
page.tags %}{{ tag }}, {% endfor %}" />
  {% elsif site.keywords %}
  <meta name="keywords" content="{{ site.keywords }}"
/>
  {% endif %}
  <meta property="og:site_name" content="{{ site.title
}}" />
</head>
```

Общий блок HEADER

```
<header class="mui-appbar-height site-appbar mui-z2">
  <div class="mui-container">
      <div style="float: left">
    <a class="site-title" href="{{ site.baseurl }}/">
        <h1>{{ site.title }}</h1>
    </a>
  </div>
  </div>
</header>
```

Общий блок FOOTER

```
<footer class="site-footer">
    <div class="footer-col-wrapper">
      <h3> {{site.title}}</h3>
      <p>Саранск 2019</p>
    </div>
</footer>
```

Боковое меню

```
<div class="mui-col-md-4">
  <div class="mui-panel">
```

```
    <h2>Ссылки</h2>
    <ul class="about-links">
<li><a href="https://jekyllrb.com/" class="mui-btn">О
сайте</a></li>
    </ul>
    <h2>Контакты</h2>
    <ul class="about-links">
         <li> <a href="https://vk.com/id146341441"
target="_blank" class="mui-btn"> <img
src="https://avatars.mds.yandex.net/get-
pdb/985144/590a8a98-c791-44a9-a5d7-
4791ce3353b8/s1200?webp=false" style="height:20px;
width:20px;"> VK</a></li></ul></div></div>
```

Главный файл CSS

```
---
# Only the main Sass file needs front matter (the
dashes are enough)
---
@charset "utf-8";



// Width of the content area
$content-width:    700px;

// Using media queries with like this:
// @include media-query($on-palm) {
//     .wrapper {
//         padding-right: $spacing-unit / 2;
//         padding-left: $spacing-unit / 2;
//     }
// }
@mixin media-query($device) {
    @media screen and (max-width: $device) {
        @content;
    }
}


// Import partials from `sass_dir` (defaults to
`_sass`)
```

Продолжение ПРИЛОЖЕНИЯ А

```
@import
        "colors",
        "mui",
        "syntax-highlighting"
;

body {
    padding-top: 93px;
}

i.fa {
    margin-right: 5px;
}

.mui-container {
min-width: 750px;
}

.site-appbar {
    background-color: mui-color("indigo", "500");
    left: 0;
    margin-bottom: 15px;
    overflow: auto;
    padding-left: 10px;
    padding-top: 10px;
    position: fixed;
    right: 0;
    top: 0;
    z-index: 6;
}

.site-nav {
    float: right;
    list-style: none;
    margin: 0;
    padding: 0;

    li {
        display: inline-block;

        a {
            color: #fff;
        }
```

```
    }
}

.mui-container-fluid {
    &.post-content {
        margin: 0 auto;
        max-width: $content-width;
    }

    img {
        display: block;
        max-width: 100%;
}
}
.post-meta {
    font-size: 18px;
    font-weight: bold;
}

.sharer {
    float: right;

    .sh-fb {
        color: #4A65A0;
    }
    .sh-twt {
        color: #0088cc;
    }
}

.post-content {
    position: relative;
    top: 135px;
    z-index: 3;

    .post-meta {
        padding: 10px 0 10px 5px;
    }

    .article-title {
        font-size: 45px;
        font-weight: bold;
```

```
 line-height: 50px;
        padding-bottom: 10px;
        transition: all 0.3s ease-in;

        a {
            color: #111;

            &:hover {
                color: #08c;
                text-decoration: none;
            }
        }
    }
}
.site-title {

    h1 {
        color: #FFF;
        display: inline-block;
        margin: 0;
    }
}

.post-list {
    list-style: none;
    margin: 0;
    padding: 0;
    padding-top: 15px;
}

.post-data {
    padding: 15px;
}

article {
    &.mui-panel {
        padding: 0px;
    }
}

.mui-panel {
    .post-list-title {
        margin-top: 0;
```

```
    }
}


.bg-holder {
    -moz-background-size: cover;
    -o-background-size: cover;
    -webkit-background-size: cover;
    background-color: mui-color('blue-grey', '500');
    background-size: cover;
    height: 450px;
    position: relative;
    a {
        color: #fff;

        &.overlay {
            bottom: 0;
            color: #fff;
            display: block;
            left: 0;
            position: absolute;
            right: 0;
            top: 0;
            z-index: 4;
        }
    }

    .img-overlay {
        bottom: 0;
        background-color: #000;

        display: block;
        left: 0;
        opacity: 0.5;
        position: absolute;
        right: 0;
        top: 0;
        z-index: 2;


    }

    h2 {
```

```
        bottom: 15px;
        font-size: 34px;
        font-weight: bold;
        left: 15px;
        line-height: 40px;
        right: 15px;
        margin: 0;
        position: absolute;
        z-index: 3;

    }
}

.post-full {
    position: relative;

    .bg-holder {
        left: 0;
        min-height: 450px;
        position: fixed;
        right: 0;
        top: 0;

        h1 {
            bottom: 25px;
            line-height: 50px;
        }

        a.overlay {

            position: fixed;
            background: #414040;
            padding: 5px;
            top: 15px;
            left: 15px;
            bottom: initial;
            right: initial;
            border-radius: 2px;

            i {
                margin: 0;
            }
        }
```

```
        a:hover {
            text-decoration : none;
        }
    }
}

.post-excerpt {
    font-size: 16px;
}
.post-tag {
    background-color: mui-color('indigo','400');
    color: #fff;
    display: inline-block;
    margin-bottom: 1px;
    padding: 2px 4px;
}

.post-page {
    h1 {
        bottom: 15px;
        font-size: 48px;
        font-weight: bold;
        left: 0;
        margin: 0;
        position: absolute;
        text-align: center;
        width: 100%;
        z-index: 3;
    }
}

.post-html {
    border-top: 1px solid mui-color('blue-grey','400');
    font-size: 18px;
    padding-top: 20px;

    p {
        margin-bottom: 20px;
    }

    h1, h2, h3, h4, h5, h6 {
        font-weight: bold;
    }
```

```
   pre {
        @extend .mui-panel;

        code {
            color: initial;
        }
    }
    code {
        color: mui-color('red', '500');
    }

    blockquote {
    border-left: 2px solid mui-color('teal', '400');
        margin: 0;
        padding-left: 15px;
    }
    img {
        width: 50px;

    }
}

article {
    footer {
        border-top: 1px solid mui-color('grey', '500');
        font-size: 18px;
        padding: 20px 0;
        text-align: center;
    }
}

.paginator {
    text-align: center;
}
.about-links {
    list-style: none;
    margin: 0;
    padding: 0;

    .mui-btn {
        display: block;
        text-align: left;
    }
```

```
}
.site-footer {
    text-align: center;
}

.bg-grey {
    background-color: mui-color('grey', '500');
}
.bg-orange {
   background-color: mui-color('orange', '500');
    a {
        color: #000;

    }
}
.bg-yellow {
    background-color: mui-color('yellow', '500');
    a {
        color: #000;
    }
}
.bg-amber {
    background-color: mui-color('amber', '500');
    a {
        color: #000;
    }
}
.bg-light-blue {
    background-color: mui-color('light-blue', '500');
}
.bg-lime {
    background-color: mui-color('lime', '500');
    a {
        color: #000;
    }
}
.bg-teal {
    background-color: mui-color('teal', '500');
}
.bg-blue {
    background-color: mui-color('blue', '500');
}
.bg-black-87 {
```

```
   background-color: mui-color('black-alpha-87');
}
.bg-blue-grey {
    background-color: mui-color('blue-grey', '500');
}
.bg-brown {
    background-color: mui-color('brown', '500');
}
.bg-indigo {
    background-color: mui-color('indigo', '500');
}
.bg-purple {
    background-color: mui-color('purple', '500');
}
.bg-deep-purple {
    background-color: mui-color('deep-purple', '500');
}
.bg-red {
    background-color: mui-color('red', '500');
}
.bg-pink {
    background-color: mui-color('pink', '500');
}

@media (max-width: 767px) {
    .bg-holder {
    background-attachment: fixed;
        h2 {
            font-size: 25px;
        }
    }
    .post-full {
        .bg-holder {
            min-height: 200px;

            h1 {
                bottom: 15px;
                font-size: 32px;
                line-height: 32px;
            }
            a.overlay {
                left: initial;
                right: 15px;
```

```
            }
        }
    }
    .sharer {
        display: block;
        float: none;
    }
}

@media (max-width: 480px) {
ody {
        padding-top: 82px;
    }
    .post-full {
        .bg-holder {
            min-height: 300px;
        }
    }
}
```

Отзыв на бакалаврскую работу

«Создание статических веб-сайтов с помощью генератора Jekyll»

студента 4 курса факультета математики и информационных технологий

А.Ю. Солдаткина

В данной бакалаврской работе была поставлена задача изучения современных технологий для генерации статических веб-сайтов, состоящих из готовых HTML-страниц и не требующих для своей работы серверных сценариев или баз данных.

Автор работы разработал статический сайт для новостного блога, который формируется с помощью генератора Jekyll, и разместил его на бесплатном хостинге GitHub Pages. К блогу были добавлены дополнительные интерактивные возможности с помощью сторонних сервисов.

В целом бакалаврская работа соответствует заявленной теме и раскрывает исследуемые технологии. В ходе выполнения бакалаврской работы А.Ю. Солдаткин показал себя грамотным специалистом, способным решать поставленные задачи.

Считаю, что бакалаврская работа А.Ю. Солдаткина заслуживает оценки "отлично".

Руководитель дипломной работы
к.ф. – м.н., доцент кафедры
фундаментальной информатики А.В. Попов

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
"НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
МОРДОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМ. Н.П. ОГАРЁВА"

ОТЗЫВ РЕЦЕНЗЕНТА
О ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ

Студента Солдаткина Андрея Юрьевича
Факультет математики и информационных технологий
Кафедра фундаментальной информатики и информационных технологий Группа 402
Направление подготовки Фундаментальная информатика и информационные технологии
Квалификация (степень) бакалавр
Наименование темы: Создание статических веб-сайтов с помощью генератора Jekyll
Рецензент Сухарев Л. А., МГУ им. Н.П. Огарева, кандидат физико-математических наук, доцент

ОЦЕНКА ВЫПУСКНОЙ РАБОТЫ

№ п/п	Показатели		Оценка				
			5	4	3	2	0*
1.	Актуальность тематики работы		+				
2.	Степень полноты обзора состояния вопроса и корректность постановки задачи			+			
3.	Уровень и корректность использования в работе методов исследования, математического моделирования		+				
4.	Степень комплексности работы, применение в ней знаний естественнонаучных, социально-экономических, общепрофессиональных и специальных дисциплин		+				
5.	Ясность, четкость, последовательность и обоснованность изложения		+				
6.	Применение современного математического и программного обеспечения, компьютерных технологий в работе		+				
7.	Качество оформления (общий уровень грамотности, стиль изложения, качество иллюстраций, соответствие требованиям стандарта)		+				
8.	Оригинальность и новизна полученных результатов (научных, конструкторских и технологических решений)			+			
9.	Тип работы	фундаментальная с оригинальными результатами					
		реферативная					
		прикладная	+				
10.	Рекомендации	к опубликованию					
		к внедрению	+				
ИТОГОВАЯ ОЦЕНКА			*отлично*				

* – не оценивается (трудно оценить)

Отмеченные достоинства:

Работа изложена ясно и четко, прослеживается логическая связь между разделами. Подробно описаны плюсы и минусы статических и динамических сайтов, указаны области их применения.

Работа имеет практическое применение – приемы, использованные автором для генерации новостного блога, могут использоваться при разработке других статических сайтов.

Отмеченные недостатки:

Недостаточно подробно описан механизм трансформации входных текстовых файлов в структуру веб-блога при использовании генератора статических сайтов Jekyll.

Заключение:

Указанный недостаток не умаляет достоинств работы, которая удовлетворяет требованиям к выпускным квалификационным работам. Считаю, что выполненная работа заслуживает оценки «отлично», а ее автор, Солдаткин Андрей Юрьевич, заслуживает присвоения степени бакалавра по направлению подготовки «Фундаментальная информатика и информационные технологии».

10 июня 2019 г. Рецензент _____
(подпись)

Заявление о самостоятельном характере выполнения работы

Я, Солдаткин Андрей Юрьевич, обучающийся 4 курса направления подготовки 02.03.02 – Фундаментальная информатика и информационные технологии, заявляю, что в моей работе на тему «Создание статических веб-сайтов с помощью генератора Jekyll», представленной в Государственную экзаменационную комиссию для публичной защиты, не содержится элементов неправомерных заимствований.

Все прямые заимствования из печатных и электронных источников, а также ранее защищённых письменных работ, кандидатских и докторских диссертаций имеют соответствующие ссылки.

Я ознакомлен с действующим в Университете Положением о проверке работ, обучающихся ФГБОУ ВО «МГУ им. Н.П. Огарёва» на наличие заимствований, в соответствии с которым обнаружение неправомерных заимствований является основанием для отрицательного отзыва руководителя работы.

04.06.2019

Подпись обучающегося | Дата

Работа представлена для проверки в Системе «Антиплагиат.ВУЗ»

04.06.2019

Дата представления работы | *подпись руководителя*

ОТЧЕТ
о результатах проверки работы обучающегося
на наличие заимствований

Ф.И.О. автора работы Солдаткин Андрей Юрьевич
Тема работы Создание статических веб-сайтов с помощью генератора Jekyll
Руководитель канд. физ.-мат. наук А. В. Попов

Представленная работа прошла проверку на наличие заимствований в системе «Антиплагиат.ВУЗ»

Результаты автоматической проверки: оригинальность – 86,45 %
цитирования – 0,48 %
заимствования –13,07 %

Результаты анализа полного отчета на наличие заимствований:
правомерные заимствования: да, 13,07%

корректные цитирования: да, 0,48%

неправомерные заимствования: нет

признаки обхода системы: нет

Общее заключение об итоговой оригинальности работы и возможности ее допуска к защите: Итоговая оригинальность работы составляет 86,45%, Солдаткин А. Ю. допускается к защите выпускной квалификационной работы (в форме бакалаврской работы)

Руководитель
канд. физ.-мат. наук 04.06.2019 А. В. Попов
подпись, дата

Заведующему кафедрой
фундаментальной информатики
А. Г. Смольянову
студента 4 курса
очной формы обучения
на бесплатной основе
направления подготовки
02.03.02 – Фундаментальная
информатика и информационные
технологии факультета математики
и информационных технологий
ФГБОУ ВО «МГУ им. Н.П.Огарёва»
Солдаткина Андрея Юрьевича

заявление.

Прошу разместить мою выпускную квалификационную работу на тему «Создание статических веб-сайтов с помощью генератора Jekyll» в электронной библиотечной системе университета в полном объеме.

21.06.19
Дата

Подпись