

МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
“ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ”

Факультет компьютерных наук
Кафедра Программирования и информационных технологий

Курсовой проект

*Веб-приложение для публикации и просмотра новостей
“Публичный корпоративный блог”*

09.03.04 Программная инженерия

Обучающийся _____ *Бородин А.О., 3 курс*
Обучающийся _____ *Буйлов Н.О., 3 курс*
Обучающийся _____ *Свиридов М.А., 3 курс*
Руководитель _____ *Нужных А.В., преподаватель*

Воронеж, 2020

Оглавление

Введение	4
Постановка задачи	5
1. Аналитическая часть	6
1.1. Анализ предметной области	6
1.1.1. Информирование	6
1.1.2. Продвижение	6
1.2. Анализ существующих решений	7
1.2.1. blogs.cisco.com	7
1.2.2. blog.dataart.com	10
1.2.3. habr.com	12
1.3. Анализ требований	14
1.4. Анализ продуктовых сценариев	17
1.4.1. Просмотр гостем статьи	17
1.4.2. Подписка гостей на рассылку	18
1.4.3. Авторизация авторов	20
1.4.4. Добавление автором новой статьи	21
1.5. Выбор архитектуры приложения	22
1.6. Выбор методов и средств реализации серверной части	24
1.6.1. Выбор фреймворка	24
1.6.2. Выбор СУБД	26
1.7. Выбор методов и средств реализации клиентской части	26
2. Реализация	28
2.1. Реализация серверной части приложения	28
2.1.1. Django REST Framework (DRF)	28
2.1.2. Swagger-документация	28
2.1.3. reCAPTCHA v3, Select2 и CKEditor	29
2.2. Реализация клиентской части приложения	30
2.2.1. Общие элементы всех страниц	31
2.2.2. Главная страница	33
2.2.3. Страница «О нас»	34

2.2.4. Страница «Контакты»	35
2.2.5. Страница поста	36
2.2.6. Личная страница автора	38
2.2.7. Страница редактирования личной информации	39
2.2.8. Страница добавления нового поста	40
2.2.9. Страница редактирования поста	41
2.2.10. Страница администратора со списком статей	42
2.2.11. Страница администратора со списком авторов	42
2.2.12. Страница добавления нового автора	43
2.2.13. Страница панели администратора	43
2.3. Подключение аналитики	44
Тестирование	45
Заключение	46
Дальнейшее развитие	47
Список использованных источников	48
Приложения	49

Введение

Блоги бывают корпоративными и персональными. В этом курсовом проекте мы рассмотрим процесс производства корпоративного блога. Под корпоративным блогом понимается блог, который ведётся группой людей или целой организацией.

Корпоративный блог может использоваться исключительно внутри организации — такой блог называют внутрикорпоративным. Другая разновидность блога — это публичный блог компании. В отличие от внутрикорпоративного, он нацелен на общение с широким кругом клиентов и деловой средой. В нашей работе речь пойдет как раз о публичном корпоративном блоге.

Корпоративный блог — это не просто новостная лента компании. Сама суть блога подразумевает диалог между компанией и читателями.

Постановка задачи

Цель курсового проекта – работая в команде, разработать систему для ведения публичного корпоративного блога, распространения новостей и уведомления о новых мероприятиях компании.

Система создается с целью:

- 1) упрощения информирования посетителей сайта о новостях, событиях, возможностях, а также новой продукции и ее технических характеристиках компании;
- 2) привлечения лиц, заинтересованных в продуктах и роде деятельности компании.

Для достижения поставленной целей необходимо решить следующие задачи:

1. Анализ предметной области
2. Выбор методов и средств разработки
3. Реализация приложения
4. Тестирование веб-приложения

1. Аналитическая часть

1.1. Анализ предметной области

Зачем компания заводит корпоративный блог? Блог заводят ради двух основных целей — информирования и продвижения.

Один блог может выполнять несколько функций одновременно. Рассмотрим самые популярные из них.

1.1.1. Информирование

Блог заводят, чтобы выносить наружу процессы, которые происходят внутри компании. Функции информирования:

Новостная. Людям, заинтересованным в деятельности компании, нужно сообщать о новостях и событиях, новых продуктах и возможностях.

Поддержка и обучение. Блог помогает поддерживать связь с клиентами и рассказывать о том, как пользоваться продуктом компании.

Социальная. В больших компаниях, состоящих из множества отделов, сотрудники не всегда знают, что происходит в соседних отделах. В этом случае, корпоративный блог выступает как площадка информирования и социализации.

1.1.2. Продвижение

Под продвижением можно понять действия, направленные на увеличение узнаваемости компании или продукта, повышение рыночной доли и привлечение новых клиентов.

Увеличение трафика. С помощью экспертных статей, публикуемых в блоге, компания привлекает больше трафика, который способствует увеличению продаж.

Популяризация бренда. Блог способствует рассказать о компании большому количеству людей, не тратясь на рекламу.

Привлечение целевой аудитории. Блог позволяет привлечь внимание разных типов целевой аудитории: это потенциальные клиенты (которые ещё не задумывались о покупке продукта), будущие клиенты (которые раздумывают насчет покупки) и текущие клиенты (которые уже купили продукт).

Развитие индустрии. Публикуемые в блоге статьи экспертов по профессиональной тематике дают индустрии новые знания и двигают ее вперед.

1.2. Анализ существующих решений

1.2.1. blogs.cisco.com

Корпоративный блог, написанный работниками компании, с огромным количеством информации и с удобнейшим фильтром для поиска нужной темы или статьи.

На главной странице расположен список постов, который реализован в виде плитки. Каждый пост в плитке состоит из изображения-превью, краткого описания статьи и информации об авторе. Снизу всех постов находится блок с пагинацией.

В нижней части главной страницы расположен блок с контактной информацией и разделами сайта.

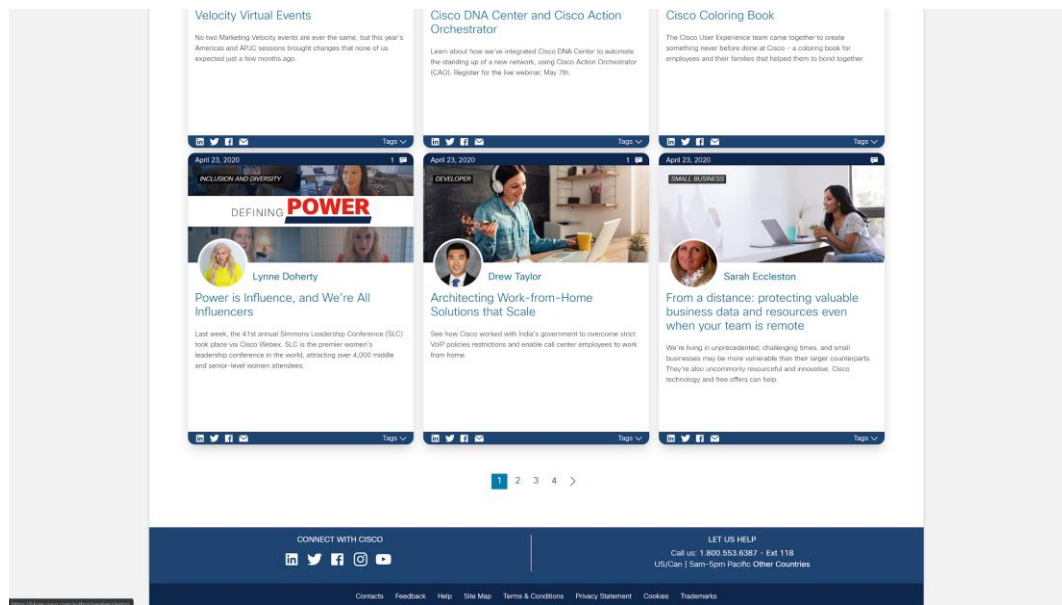


Рисунок 1. Главная страница с постами блога Cisco

В верхней части главной страницы находится блок с логотипом, меню, в котором можно выбрать отдельную категорию для фильтрации постов, а также поиском для фильтрации статей по запросу.

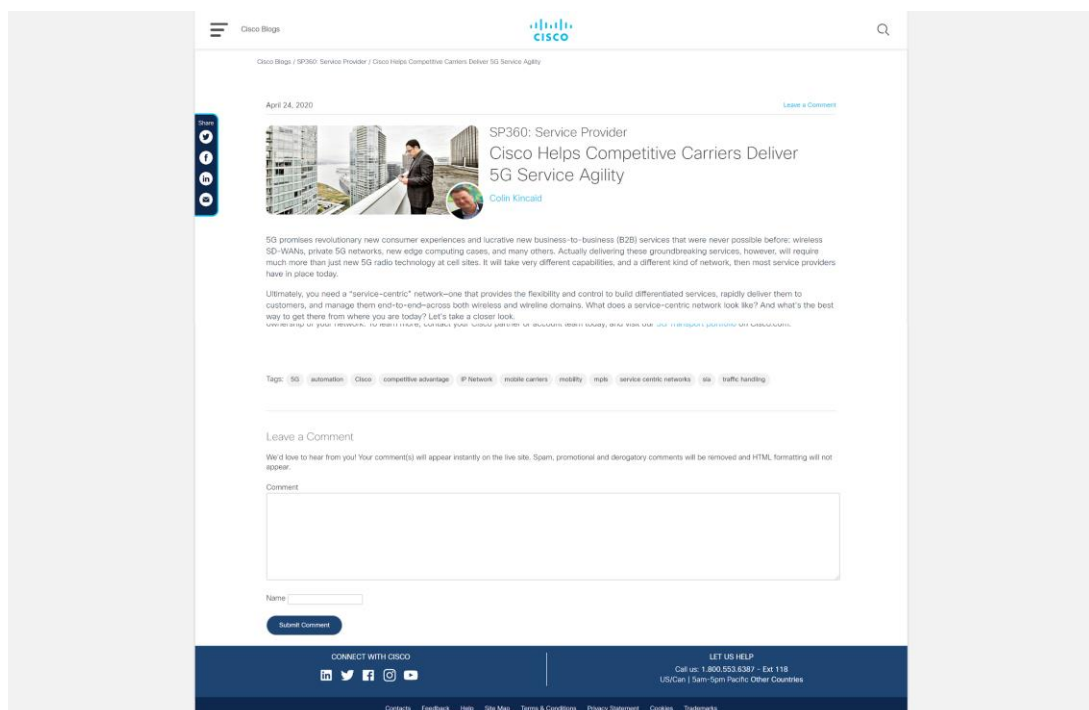


Рисунок 2. Страница с содержанием поста

На странице с постом располагается полная информация о статье. В верхней части страницы расположена информация о самой статье – картинка-превью, её заголовок, дата опубликования, ссылка на автора и его аватар, а также ссылка на быстрый переход к блоку для комментирования.

После верхнего блока располагается полный текст статьи с тегами.

В нижней части страницы расположен блок с комментариями и формой для добавления нового комментария. Форма состоит из двух полей - поле для имени и поле для текста сообщения - и кнопки отправки. Для отправки комментария не требуется регистрации, что облегчает взаимодействие с пользователями сайта.



Рисунок 3. Личная страница автора

Страница автора состоит из двух блоков - блок информации об авторе и блока с постами автора.

Блок информации об авторе состоит из фотографии автора, имени, его должности, а также краткой биографической информации.

Блок с постами автора идентичен плитке с постами на главной странице, однако здесь расположены только те посты, которые были написаны автором.

1.2.2. blog.dataart.com

Блог международной сети компаний, которые проектируют, разрабатывают, модернизируют и поддерживают IT-решения.

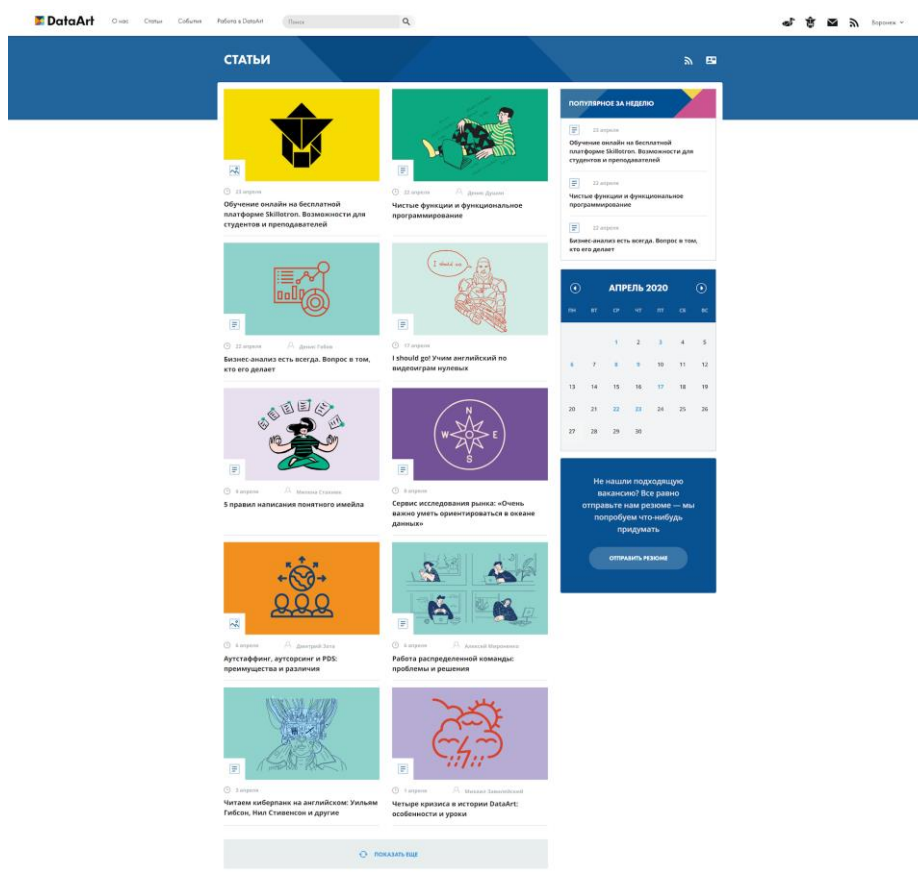


Рисунок 4. Главная страница блога компании DataArt

Главная страница блога аналогична странице блога Cisco. Главным отличием является наличие справа от списка статей блока с популярными за неделю статьями и календарем, для фильтрации статей по дате.

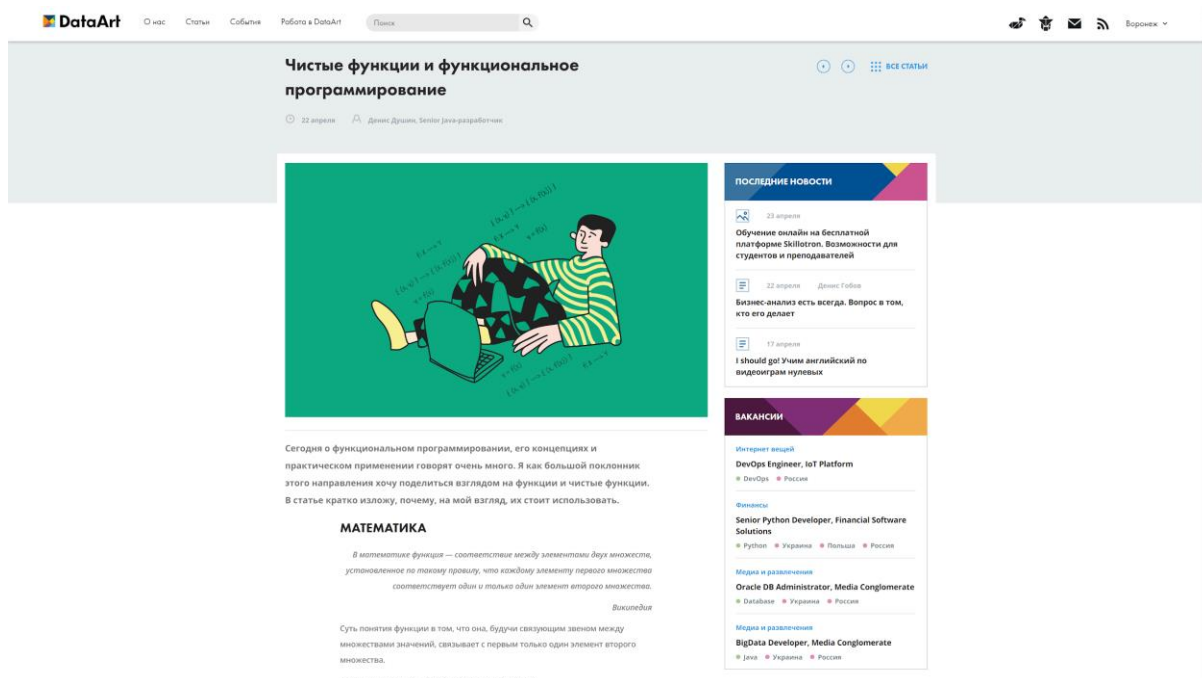


Рисунок 5. Страница поста на сайте DataArt

На странице поста справа от статьи также расположен блок с новостями, а также блок с вакансиями, доступными в данный момент. На странице отсутствует блок с комментариями для взаимодействия с пользователями сайта.

Блог не предоставляет возможности писать комментарии для статей и просматривать информацию об авторе и написанные им статьи на его личной странице, что является существенным недостатком данного сайта.

1.2.3. habr.com

Веб-сайт в формате системы тематических коллективных блогов (именуемых хабами) с элементами новостного сайта, созданный для публикации новостей, аналитических статей, мыслей, связанных с информационными технологиями, бизнесом и интернетом.

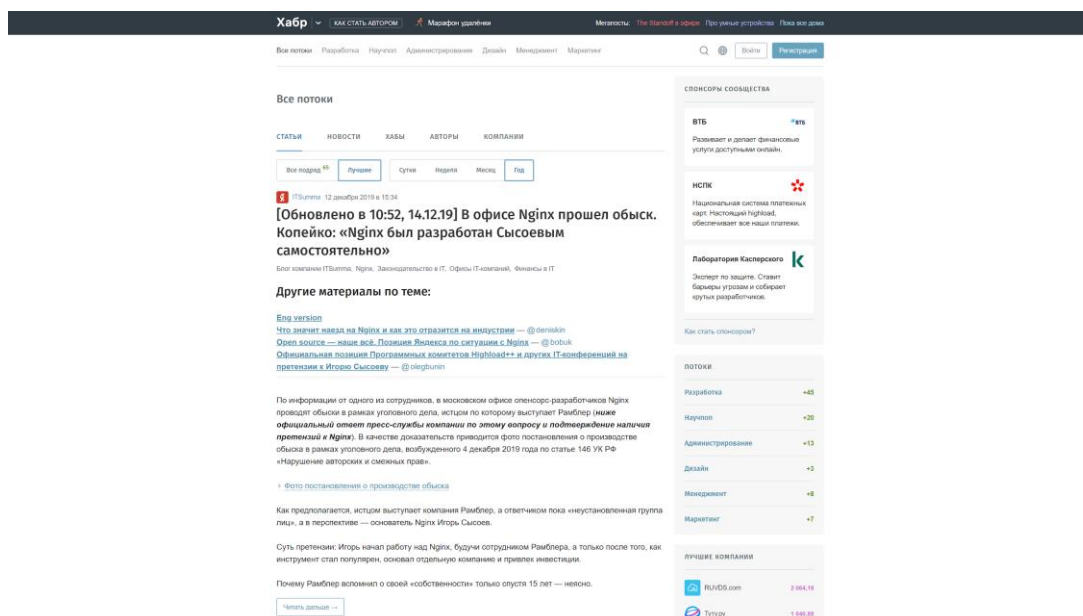


Рисунок 6. Главная страница блога habr

На главной странице блога статьи расположены в виде списка, что позволяет указать больше информации о каждом посте. На странице присутствует много способов отфильтровать список статей, так как сайт является агрегатором статей множества компаний и пользователей.

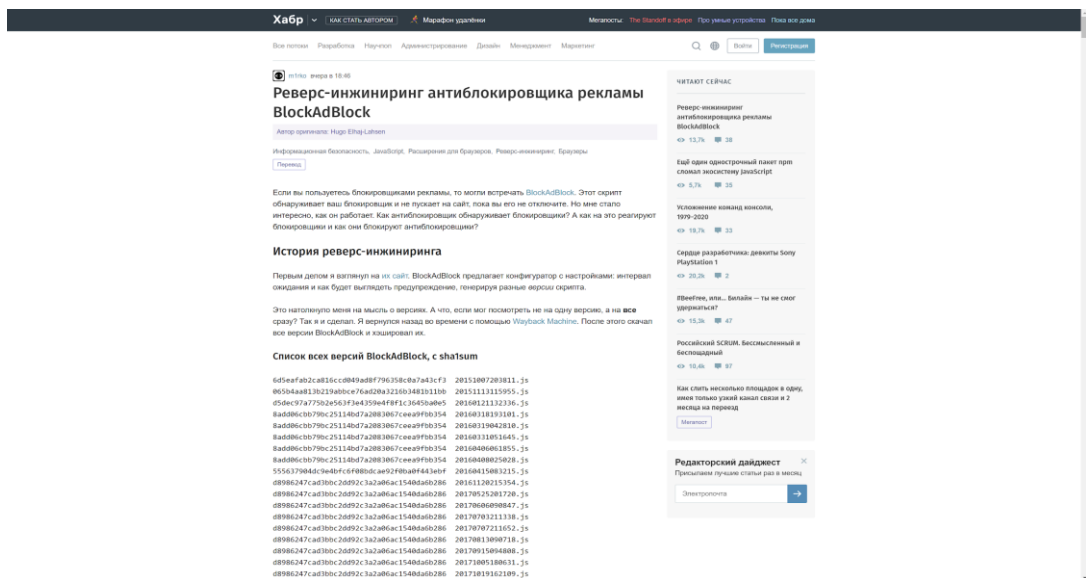


Рисунок 7. Страница с постом на habr

На странице поста справа расположены блок с подпиской по E-Mail и блок с популярными в данный момент статьями. Под содержанием статьи расположен блок с комментариями. Комментарии под постами можно оставлять только после регистрации и авторизации, что с одной стороны облегчает модерирование комментариев, с другой – усложняет взаимодействие с пользователями.

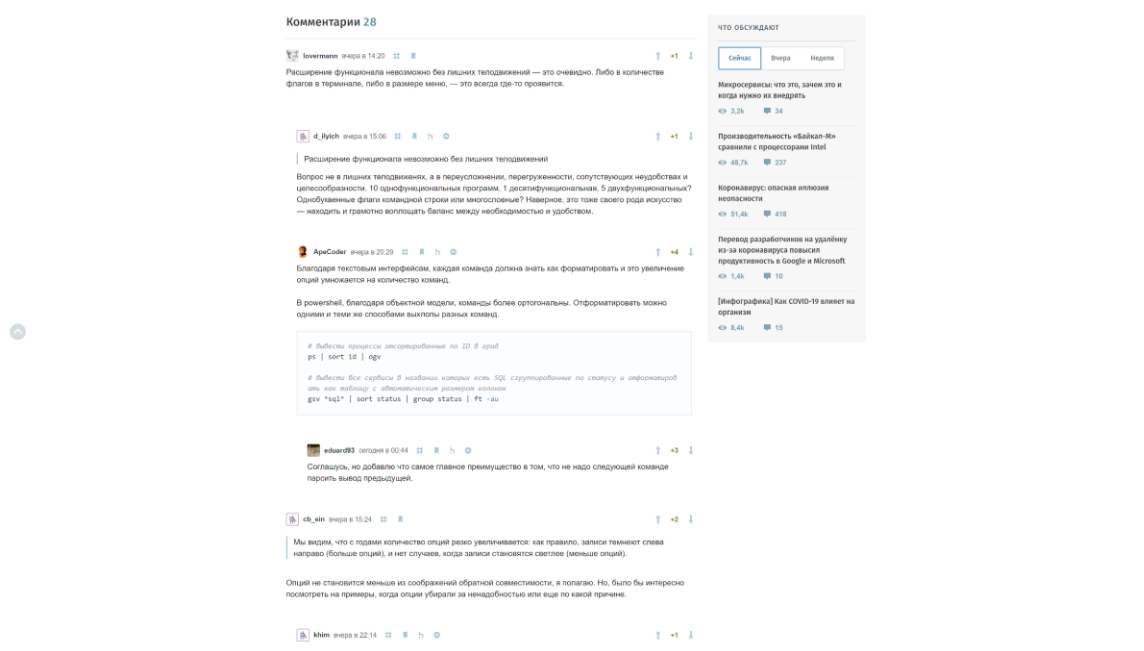


Рисунок 8. Блок с комментариями под содержанием статьи

1.3. Анализ требований

Анализ существующих решений показал, что приложение должно позволять работникам компании — авторам, публиковать статьи, редактировать их и по необходимости удалять. Также автор должен иметь возможность проводить модерацию комментариев под своими статьями и получать статистику в личном кабинете, связанную со написанными им статьями.

Для обычного пользователя приложение должно обеспечивать свободный просмотр выложенных авторами статей, комментирование их, просмотр информации о компании и возможность связаться с компанией, используя E-Mail. При просмотре статей, пользователь должен иметь возможность фильтровать список всех статей.

В системе должны быть предусмотрены три группы пользователей:

- Администратор
- Автор

- Гость

Гость обладает следующими функциями:

- Просмотр постов, выложенных авторами
- Возможность подписаться на новостную рассылку о новых статьях
- Возможность оставлять комментарии к постам
- Просмотр информации о компании
- Возможность связи с компанией путём E-Mail

Администратор, помимо функций гостя и автора, обладает следующими функциями:

- Удаление и добавление новых авторов
- Удаление любых постов
- Просмотр статистики системы
- Сброс пароля для авторов

На диаграмме вариантов использования (приложение Б №1, №2, №3, №4) отражено, какие функции должна предоставлять система каждому виду пользователей.

Таким образом, в разрабатываемой системе можно выделить следующие классы:

- Статья (Post)
- Автор (Author)
- Комментарий (Comment)
- Категория (Category)
- Подписчик (Mailing Member)

Система должна хранить об авторе следующую информацию:

- Фамилию и имя (опционально)

- Логин
- E-Mail
- Пароль
- Краткую информацию (опционально)
- Список опубликованных статей
- Фотографию (опционально)

Система должна хранить о статье следующую информацию:

- Заголовок
- Текст статьи
- Автор
- Список категорий, к которым эта статья относится
- Картинка-превью
- Дата опубликования
- Дата изменения

Система должна хранить о комментарии следующую информацию:

- Имя отправителя (опционально)
- Автор (опционально)
- Текст сообщения
- Дата опубликования
- Статья
- Комментарий-родитель

Диаграмма классов, описывающая данную систему, приведена в Приложении В. Возможные отношения между экземплярами классов приведены в диаграмме объектов (Приложение Г), а их взаимодействие и возможные сценарии использования на диаграмме взаимодействия (Приложение Д). Временные особенности взаимодействия между

объектами системы (автором и гостем) и самой системой отображены на диаграммах последовательностей (Приложения Е №1, №2, №3).

Алгоритм действий системы при реализации основных сценариев показан на диаграмме активности (Приложение Ж), а процесс изменений состояний системы при исполнении этих сценариев отражен на диаграмме состояний (Приложение И).

Система должна быть развернута на сервере, иметь базу данных и клиентскую часть. Общая структура системы представлена на диаграмме развёртывания в Приложении К.

1.4. Анализ продуктовых сценариев

В анализе продуктовых сценариев участвуют следующие пользовательские сценарии, главными целями которых являются:

- Просмотр гостем статьи
- Подписка гостем на рассылку
- Авторизация автора
- Добавление автором новой статьи

Скриншот готовых воронок из аналитики Яндекс метрики представлен в Приложении Л.

1.4.1. Просмотр гостем статьи

Для достижения поставленной цели пользователь должен:

- Перейти на главную страницу
- По желанию отфильтровать список постов с помощью категорий
- Выбрать подходящую статью
- Перейти на страницу с содержанием статьи



Рисунок 9. Воронка "Просмотр гостем статьи"

Переход на главную страницу. Стартовая точка сценария. На главной странице находится список статей, из которых пользователь может выбрать любую, с которой продолжится сценарий.

Фильтр списка статей. Пользователь должен иметь возможность каким-либо образом фильтровать список статей по категориям.

Выбор статьи. Выбор статьи пользователем может основываться на кратком описании статьи или на картинке-превью.

Переход на страницу статьи. После выбора статьи, пользователь переходит по ссылке, ведущей на страницу с полным содержанием статьи.

Данный сценарий позволяет проанализировать насколько пользователю интересно содержание сайта и какие статьи наиболее полезны для привлечения новых пользователей.

1.4.2. Подписка гостей на рассылку

Для достижения поставленной цели пользователь должен:

- Перейти на главную страницу
- Выбрать подходящую статью

- Перейти на страницу с содержанием статьи
- Успешно заполнить форму подписки справа статьи или снизу страницы
- Нажать кнопку подписаться



Рисунок 10. Воронка "Подписка гостей на рассылку"

Переход на главную страницу. Стартовая точка сценария. На главной странице находится список статей, из которых пользователь может выбрать любую, с которой продолжится сценарий.

Выбор статьи. Выбор статьи пользователем может основываться на кратком описании статьи или на картинке-превью.

Переход на страницу статьи. После выбора статьи, пользователь переходит по ссылке, ведущей на страницу с полным содержанием статьи.

Заполнение формы подписки. На странице с содержанием статьи пользователю предоставляется две формы для заполнения, которые состоят из текстового поля, в которое нужно ввести E-Mail адрес пользователя.

Нажать кнопку подписаться. Подтвердить подписку на рассылку новостей нужно нажатием на кнопку “Подписаться”, которая находится рядом с текстовым полем.

Данный сценарий позволяет проанализировать как часто пользователь подписывается на новостную рассылку, а следовательно, нравится ли пользователю контент сайта.

1.4.3. Авторизация авторов

Для достижения поставленной цели пользователь должен:

- Перейти на главную страницу
- Перейти на страницу авторизации
- Ввести данные
- Нажать кнопку “Авторизоваться”



Рисунок 11. Воронка "Авторизация авторов"

Переход на главную страницу. Стартовая точка сценария. На главной странице находится ссылка на страницу авторизации для автора.

Переход на страницу авторизации. Для входа в личный кабинет автору нужно авторизоваться в системе.

Ввести данных. Для успешной авторизации, автору нужно корректно ввести логин или E-Mail и пароль.

Нажать кнопку “Авторизоваться”. Для входа в личный кабинет после ввода данных пользователю необходимо нажать кнопку “Войти”. После нажатия пользователь попадает в личный кабинет.

Данный сценарий позволяет проанализировать активность авторов, которая выражается в написании новых статей, изменении и модерировании уже написанных статей, а также просмотре и ответе на написанные гостями комментарии.

1.4.4. Добавление автором новой статьи

Для достижения поставленной цели пользователь должен:

- Перейти на страницу добавления новой статьи
- Ввести данные статьи
- Нажать кнопку “Опубликовать”



Рисунок 12. Воронка "Добавление автором новой статьи"

Переход на страницу добавления статьи. Для опубликования новой статьи пользователю необходимо перейти на страницу добавления новой статьи. Для этого пользователю нужно нажать на специальную кнопку.

Ввести данные статьи. На странице с добавлением новой статьи пользователю предоставляется форма для заполнения, которая состоит из текстового поля для заголовка статьи, текстового поля для ввода текста статьи, списка для выбора категорий, поля для загрузки картинки-превью.

Нажать кнопку опубликовать. Подтвердить опубликование статьи нужно нажатием на кнопку “Опубликовать”, которая находится после всех полей формы для добавления статьи.

Данный сценарий позволяет проанализировать как часто авторы добавляют статьи и понять, достаточно ли система удовлетворяет потребностям авторов

1.5. Выбор архитектуры приложения

При разработке приложения был построена архитектура с использованием паттерна проектирования MVT.

Паттерн MVT (Model-View-Template) является модификацией MVC. Основное различие между этими двумя шаблонами состоит в том, что Django сам заботится о части контроллера (программный код, который контролирует взаимодействие между моделью и представлением). Шаблон представляет собой файл HTML, смешанный с языком шаблонов. Схематично паттерн можно определить следующим образом:

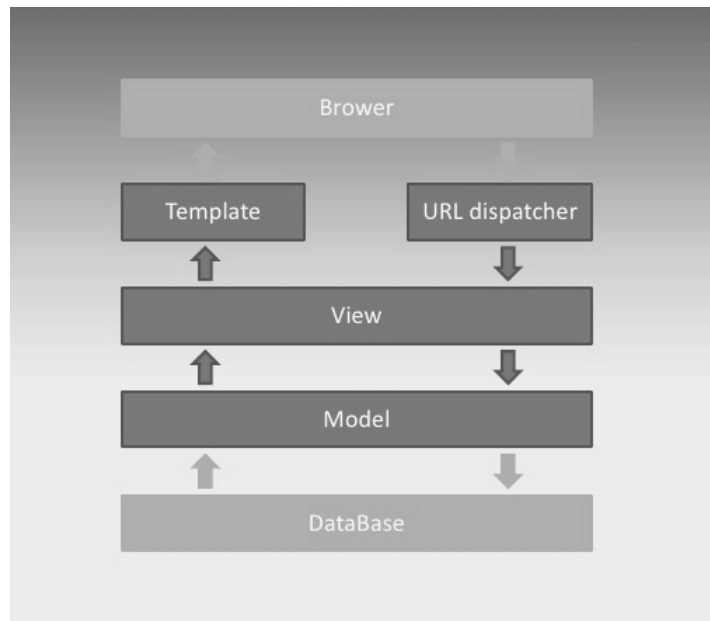


Рисунок 13. Структура паттерна MVT

Описание основных элементов паттерна MVT:

- **URL dispatcher**: при получении запроса на основании запрошенного адреса URL определяет, какой ресурс должен обрабатывать данный запрос.

- **View** получает запрос, обрабатывает его и отправляет в ответ пользователю некоторый ответ. Если для обработки запроса необходимо обращение к модели и базе данных, то View взаимодействует с ними. Для создания ответа может применять Template или шаблоны. В архитектуре MVC этому компоненту соответствуют контроллеры (но не представления).

- **Model** описывает данные, используемые в приложении. Отдельные классы, как правило, соответствуют таблицам в базе данных.

- **Template** представляет логику представления в виде сгенерированной разметки HTML. В MVC этому компоненту соответствует View, то есть представления.

Когда к приложению приходит запрос, то URL dispatcher определяет, с каким ресурсом сопоставляется данный запрос и передает этот запрос

выбранному ресурсу. Ресурс фактически представляет функцию или View, который получает запрос и определенным образом обрабатывает его. В процессе обработки View может обращаться к моделям и базе данных, получать из нее данные, или, наоборот, сохранять в нее данные. Результат обработки запроса отправляется обратно, и этот результат пользователь видит в своем браузере. Как правило, результат обработки запроса представляет сгенерированный html-код, для генерации которого применяются шаблоны (Template).

Выбор данного шаблона проектирования был обусловлен фреймворком Django, который используется для разработки системы. Такая архитектура позволяет Django успешно решать разные задачи. Выбранный шаблон проектирования позволяет четко разделить логику приложения, что облегчит поддержку и тестирование кода.

1.6. Выбор методов и средств реализации серверной части

1.6.1. Выбор фреймворка

Как было сказано выше, при реализации архитектуры системы был выбран фреймворк Django. Django в формальной форме часто описывают как «веб-фреймворк для перфекционистов с дедлайнами». Изначально он был создан для того, чтобы переходить от прототипов к готовым сервисам как можно быстрее – это замечательно подходит нам при разработке системы в условиях временной ограниченности.

Система приложений в Django позволяет разделить проект на несколько приложений и работать с ними параллельно. Такой подход позволяет с легкостью интегрировать готовые решения-приложения, что в очень сильно ускоряет и упрощает разработку.

При создании системы с помощью Django не требуется изобретать велосипед, а многие функции работают без подключения различных библиотек: Django поддерживает ORM, миграции базы данных, аутентификацию пользователя, панель администратора и формы, которые изначально недоступны в микро-фреймворках. Кроме того, Django по умолчанию безопасен и включает механизмы предотвращения распространённых атак вроде SQL-инъекций и подделки межсайтовых запросов (CSRF).

Django REST Framework является популярной библиотекой для построения API. Она имеет модульную и настраиваемую архитектуру, которая хорошо работает для создания как простых, так и сложных API.

Однако у Django имеется и ряд недостатков. Django медленно развивается из-за его масштабируемости и монолитности. Это позволяет сообществу разрабатывать сотни универсальных приложений, но снижает скорость разработки самого Django.

Кроме того, Django ORM значительно уступает последней версии SQLAlchemy. Хотя Django ORM не так гибок, как SQLAlchemy, а большая экосистема многократно используемых модулей и приложений замедляет развитие инфраструктуры, очевидно, Django является первым кандидатом на роль фреймворка для разработки на Python.

Альтернативные легкие фреймворки типа Flask, хотя и позволяют быть свободнее Django в экосистеме и конфигурации, могут потребовать лишнего времени на поиск и создание дополнительных библиотек и функциональных возможностей в долгосрочной перспективе.

1.6.2. Выбор СУБД

Из бесплатных СУБД, не ограничивающих максимальных размер базы данных, наиболее популярными являются MySQL, PostgreSQL и MongoDB. MongoDB для нашей системы не подходит, так как информация в системе структурирована и требуется соблюдение целостности данных. Что касается MySQL и PostgreSQL – обе СУБД активно поддерживаются и имеют инструменты для работы с многими фреймворками. В каких-то ситуациях MySQL более производительна, в как-то – более PostgreSQL. К тому же, выбор СУБД зачастую основан на опыте команды, что в нашем случае не является главным критерием. Но в силу популярности MySQL проще найти сотрудника с большим опытом. Кроме того, MySQL менее сложна в изучении, чем PostgreSQL. Таким образом, в качестве СУБД использовалась MySQL.

1.7. Выбор методов и средств реализации клиентской части

Клиентская часть приложения реализуется в виде веб-страниц, через которые пользователь может взаимодействовать с системой.

Для реализации клиентской части было принято решение использовать язык HTML в связке с CSS и JavaScript. Для адаптивности и упрощения процесса верстки веб-страниц используется веб-фреймворк Bootstrap.

Язык HTML был выбран из-за того, что он поддерживается всеми современными браузерами.

CSS является единственным способом задания цветов, шрифтов, стилей, расположения отдельных блоков и других аспектов представления внешнего вида веб-страниц.

JavaScript был выбран в качестве языка для реализации мелких функций системы, которые не требуют отправки запроса на сервер, потому что он поддерживается всеми браузерами.

2. Реализация

2.1. Реализация серверной части приложения

Специфика фреймворка Django предполагает наличие множества сторонних модулей, расширяющих его базовые возможности. Для упрощения и ускорения процесса разработки, нами были использованы различные сторонние библиотеки для Django. Рассмотрим наиболее важные из них, в значительной мере влияющие на функционал серверной части приложения.

2.1.1. Django REST Framework (DRF)

Для создания гибкого и мощного API для проекта мы использовали библиотеку DRF, которая работает со стандартными моделями Django.

Архитектура DRF состоит из 3-х слоёв: сериализатора, вида и маршрутизатора.

Классы сериализаторов преобразуют информацию, хранящуюся в базе данных и определенную с помощью моделей Django, в формат, который легко и эффективно передается через API.

Вид, или `ViewSet`, определяет функции (чтение, создание, обновление, удаление), которые будут доступны через API.

Маршрутизатор определяет URL-адреса, которые будут предоставлять доступ к каждому виду.

2.1.2. Swagger-документация

Документирование API приложение ведётся с помощью ПО с открытым исходным кодом Swagger. Этот инструмент позволяет

разработчикам вести и предоставлять документацию RESTful веб-сервисов с возможностью тестирования запросов к методам API через Swagger UI.

Для автоматического документирования API DRF, нами была использована библиотека `drf_yasg`, которая самостоятельно генерирует схемы Swagger и Redoc на основании моделей Django.

2.1.3. reCAPTCHA v3, Select2 и CKEditor

Для начальной защиты сайта от спама было решено использовать систему reCAPTCHA. Библиотека `django-recaptcha3` позволяет легко интегрировать поле reCaptcha на странице путём добавления специального поля в формах Django и использования специальных тэгов в файлах шаблонах Django, а настройки капчи вынести в файл настроек Django.

Select2 — плагин, который дает возможность настраивать блоки выбора из списка, а также имеет встроенный динамический поиск данных с внутренней прокруткой. Для простой интеграции Select2 в Django нами используется библиотека `django-select2`, с помощью которой в формы Django добавляются поля с виджетом Select2, а настройки Select2 выносятся в файл настроек Django.

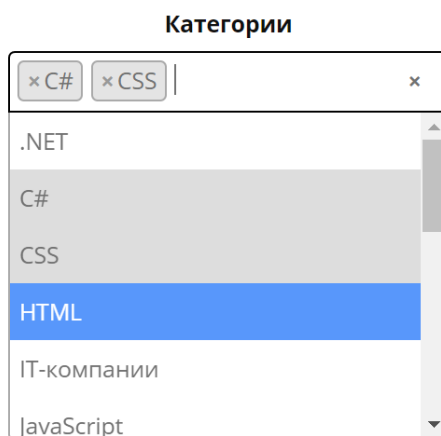


Рисунок 14. Пример выпадающего списка Select2

CKEditor – это готовый для использования текстовый редактор HTML, созданный для упрощения создания содержания веб-страниц. Для интеграции CKEditor в наш проект используется библиотека `django-sckeditor`, которая позволяет изменять обычные текстовые формы, определённые в моделях и формах Django, на текстовые формы с расширенным функционалом, которые могут быть изменены по желанию.

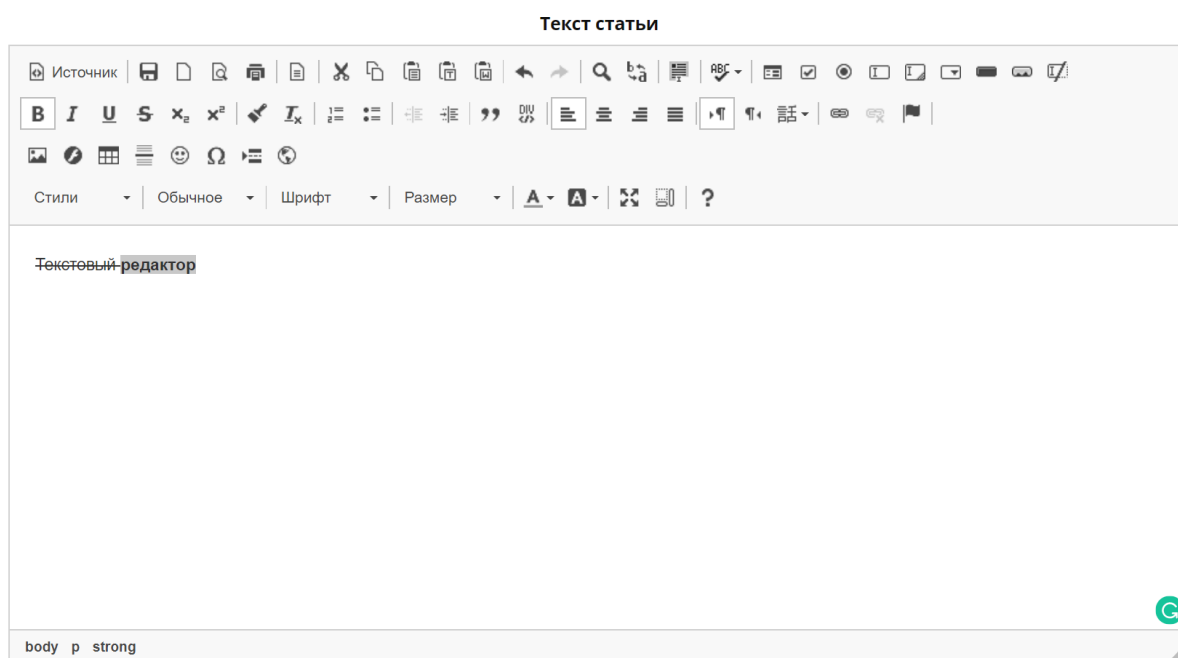


Рисунок 15. Использование CKEditor для текста статьи

2.2. Реализация клиентской части приложения

При реализации клиентской части приложения использовался адаптивный (адаптирован под десктопы, смартфоны и планшеты) мультифункциональный шаблон HTML/CSS шаблон Blak.

Он поставляется с более чем 20 вариантами домашних страниц и сотней других элементов и блоков. Строгий и стильный чёрно-белый дизайн хорошо подошёл под тематику нашего корпоративного блога.

Шаблон предоставляет более 10 готовых страниц для блога с различными расположениями блоков. Готовые страницы были изменены и скорректированы нами под необходимый макет.

2.2.1. Общие элементы всех страниц

В верхней части страницы – хедере – слева расположено название компании, а справа – блок ссылок с основной навигацией по сайту. Набор этих ссылок зависит от роли пользователя.

Если пользователь является гостем или автором, в блоке расположены ссылки на главную страницу с постами, страницу с информацией о компании и страницу с возможностью отправления компании электронного письма.

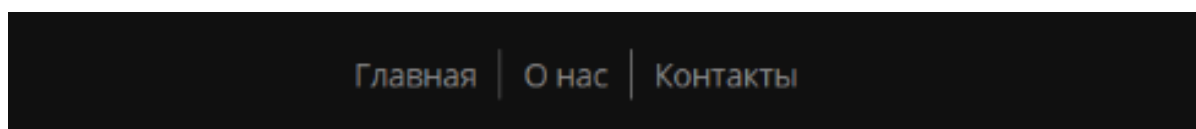


Рисунок 16. Ссылки если пользователь гость или автор

Если пользователем является администратор, то в блоке, помимо этих ссылок, ещё добавляются ссылки на страницу панели администратора на сайте, на которой он может увидеть информацию о сайте, а также ссылка на встроенную в Django панель администратора, с помощью которой он может изменять записи в базе данных.

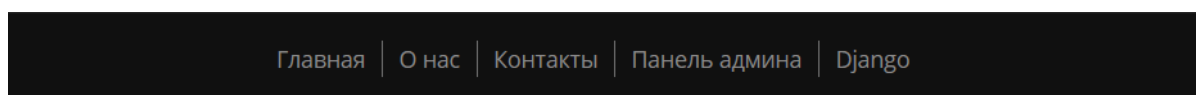


Рисунок 17. Ссылки, если пользователь администратор

Ниже панели с ссылками расположен блок с картинкой с названием страницы, которые могут изменяться в зависимости от страницы, на которой находится пользователь.

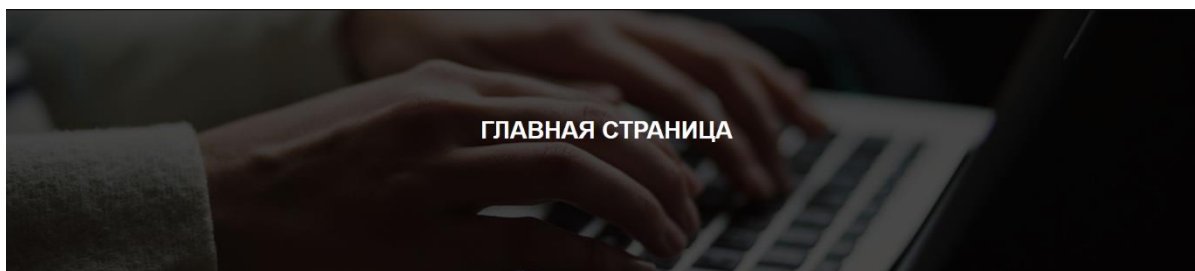


Рисунок 18. Блок с картинкой

Ниже блока с картинкой располагается вспомогательный блок с набором ссылок для быстрого перехода на различные страницы. Набор ссылок в блок также зависит от роли пользователя.

Если пользователь является гостем, то в блоке ссылок находится одна ссылка, ведущая на страницу авторизации на сайте для авторов и администраторов.

БЛОГ КОМПАНИИ NBZDR

Авторизация

Рисунок 19. Вспомогательные ссылки гостя

Если пользователь является автором, то в блоке ссылок находятся ссылка на личную страницу автора, ссылка на страницу добавления нового поста и ссылка на выход из личного кабинета. Если пользователь расположен на личной странице, то в блок добавляется ссылка, ведущая на страницу редактирования личной информации.

БЛОГ КОМПАНИИ NBZDR

Личная страница » Добавить пост » Редактировать » Выход

Рисунок 20. Вспомогательные ссылки автора

Если пользователь является администратором, то в блоке расположены ссылка на страницу со списком всех постов, ссылка на страницу со списком всех авторов, ссылка для перехода на страницу добавления нового автора и ссылка на выход из личного кабинета.

Рисунок 21. Вспомогательные ссылки администратора

В нижней части страниц – футере – расположен блок с формой для подписки на обновление контента сайта, заполнив которую и нажав на кнопку подписаться, введённый E-Mail будет добавлен в систему, и при добавлении новой статьи, на него будет выслано оповещение.

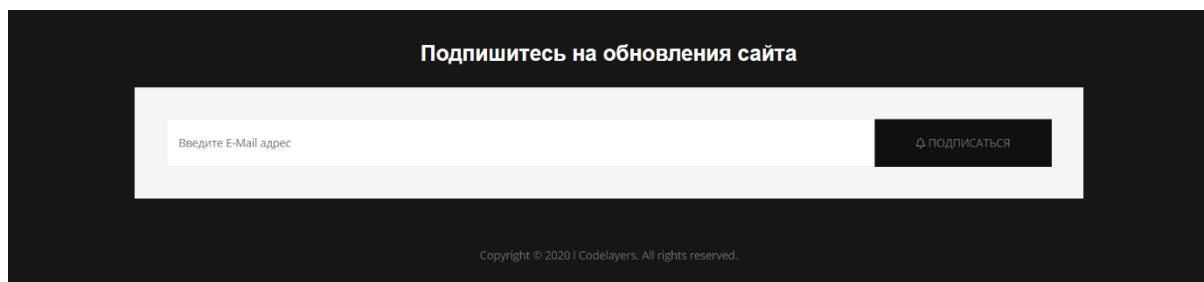


Рисунок 22. Блок с формой подписки на обновления

2.2.2. Главная страница

На главной странице расположен список постов, снизу которого находится блок с пагинацией. Слева от списка постов находится блок для фильтрации статей по категориям, с помощью которого пользователь может отобразить посты, относящиеся к конкретным категориям.

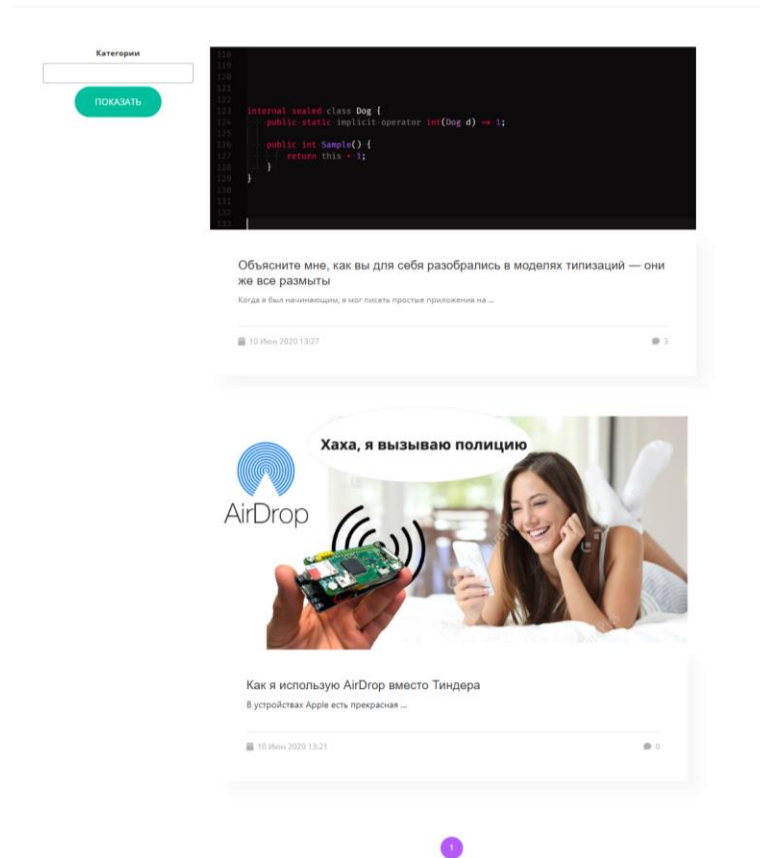


Рисунок 23. Блок со списком постов, фильтром категорий и пагинацией

2.2.3. Страница «О нас»

На странице «О нас» приведена краткая информация о каждом члене команды и ссылки на их социальные сети.

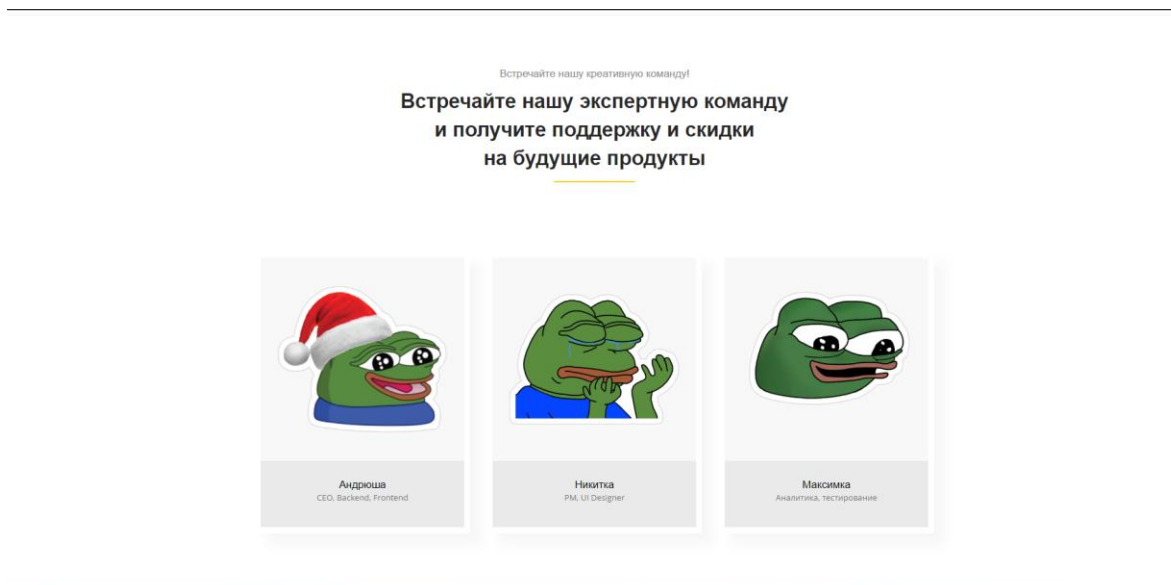


Рисунок 24. Блок с информацией о команде

2.2.4. Страница «Контакты»

На странице «Контакты» находится форма для отправки сообщения, а также общая краткая информация о команде разработчиков, расположенная справа от контактной формы.

Имя *

Введите имя

E-mail *

Введите e-mail

Тема *

Введите тему сообщения

Сообщение *

Введите сообщение

Отправить сообщение

Свяжитесь с нами!

Информация об адресе

Nebezdari Team

13031 W Jefferson Blvd UNIT 200, Los Angeles, CA 90094

Телефон: +7 952 952 52 38

E-mail: admin@nebezdari.ru


Веб-сайт: www.nebezdari.ru

Рисунок 25. Блок с формой для отправки электронного письма

2.2.5. Страница поста

Страница поста состоит из блока с информацией, касающейся статьи. В этом блоке находится информация об авторе (его имя, фамилия и аватар), название статьи, список категорий, к которым эта статья относится, дата публикации и непосредственно текст статьи.

В правой части страницы расположен блок с подпиской, аналогичный тому, который находится внизу страницы, а также список других статей для быстрого перехода к другому посту.



ANDREI BORODIN

Объясните мне, как вы для себя разобрались в моделях типизаций — они же все размыты

JavaScript | Программирование | .NET | C# | TypeScript
Июнь 10, 2020, 1:27 п.п.

Когда я был начинающим, я мог писать простые приложения на C# и C++. Долго игрался с консольными прогами, пощупал десктопные, и в какой-то момент захотел сделать сайт. Меня ждал большой сюрприз — чтобы делать сайты, одного сишарпа мало. Надо ещё знать жс, хтмл, цсс и прочую фронттовую хрень. Я потратил около недели на эти вещи, и понял — не мое. Я мог написать какой то код на джаваскрипт, но он не содержал типов, и я никак не мог взять в толк — как к этому вообще подходить. Это какое-то игрушечное программирование. Ну и забросил к чертям.

Уже потом, работе на третьей, меня перевели в отдел, где делали веб. Я подумывал уволиться, но мне объяснили — там тайпскрипт, тайпскрипт — это такой сишарп для браузера.

Я согласился, изучил его, и сейчас это один из моих любимых ЯП. Но. Тайпскрипт — это вот вообще не сишарп. Это язык с принципиально другой системой типов. Сложной, мощной, но другой.

Самих параметров типизаций несколько.

Есть статическая/динамическая типизация. Статическая — это когда типы данных известны на этапе компиляции. Компилятор знает типы переменных, и на их основании проверяет корректность программы. Динамическая — это когда типы известны только на стадии выполнения, и компилятор при сборке не проверяет ничего.

При этом все статически типизированные ЯП, которые я использовал, дают возможности для динамической типизации. Any в TS, Dynamic в сишарпе. В конце концов тот же тип Object — по идее, я запикиваю в него все что угодно, и говорю, что у меня статически типизированный код. Но строго говоря, это не совсем так. Потому что я могу принять Object извне, и покастить его к чему угодно, не делая никаких проверок — это ли не элемент дин типизации?

E-Mail адрес

ПОДПИСАТЬСЯ

ДРУГИЕ ПОСТЫ

КАК Я ИСПОЛЗУЮ AIRDROP ВМЕСТО ТИНДЕРА
Andrei Borodin 0

ОБЪЯСНИТЕ МНЕ, КАК ВЫ ДЛЯ СЕБЯ РАЗОБРАЛИСЬ В МОДЕЛЯХ ТИПИЗАЦИЙ — ОНИ ЖЕ ВСЕ РАЗМЫТЫ
Andrei Borodin 3

Рисунок 26. Блок с текстом статьи

Снизу блока с содержанием статьи расположен блок со списком всех комментариев к данной статье, а также блок с формой оставления нового комментария. Пользователь имеет возможность ответить на определённый комментарий, нажимая на кнопку «Ответить», находящуюся рядом с каждым комментарием.

3 Комментариев

Drag13 #

Июнь 10, 2020, 1:33 п.п.

Наконец то не про "боль и страдания короля", а про что-то интересное с технической точки зрения.

↩ Ответить

AlexJameson #

Июнь 10, 2020, 1:33 п.п.

Да, интересно и раскрыта объемная тема в простых словах. Но для меня главный вопрос — куда делась предыдущая статья про F#?

↩ Ответить

dendy (Andrei Borodin) #

Июнь 10, 2020, 1:35 п.п.

Забрал переписывать, что бы донести до вас то же самое посильнее

↩ Ответить

Рисунок 27. Блок с комментариями

Опубликовать комментарий!

Имя

Текст сообщения

ОПУБЛИКОВАТЬ КОММЕНТАРИЙ

Рисунок 28. Форма для отправки комментария

2.2.6. Личная страница автора

На личной странице автора расположен блок с личной информацией об авторе – его имя, фамилия, аватар и краткое описание его биографии. Ниже этого блока находится список последних постов автора с их кратким содержанием, отсортированный по дате и времени опубликования. Если на странице находится её владелец, то он имеет возможность перехода на страницу редактирования или удаления статьи с помощью кнопок, расположенных рядом с каждой статьёй в списке всех статей.

БЛОГ КОМПАНИИ NBZDR

Личная страница » Добавить пост » Редактировать » Выход



Andrei Borodin

Если я тебе не нравлюсь - застрелись, я не исправлюсь

Рисунок 29. Блок с личной информацией об авторе

Последние посты автора

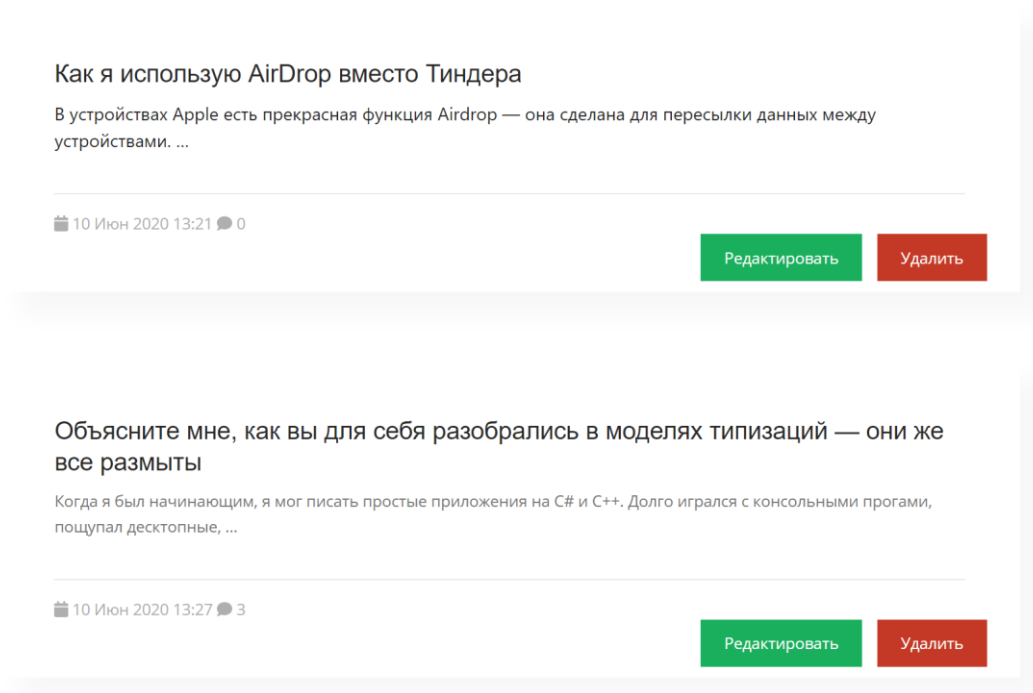


Рисунок 30. Блок со списком постов автора

2.2.7. Страница редактирования личной информации

На этой странице автор может изменить свои имя, фамилию, аватар, и указать краткую информацию с помощью текстового поля.

На данный момент: avatars/pepeEZ.jpg

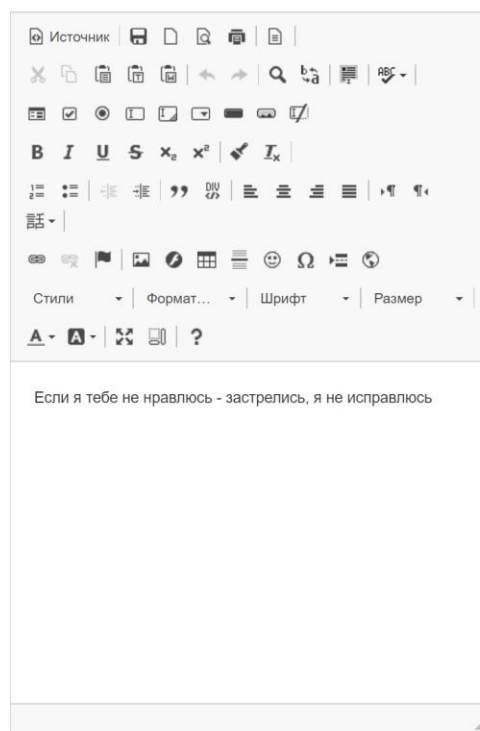
Изменить:

Выберите файл

Файл не выбран

Andrei

Borodin



СОХРАНИТЬ

Рисунок 31. Редактирование личной информации автором

2.2.8. Страница добавления нового поста

На странице добавления нового поста находится форма для добавления статьи, заполнив которую и нажав на кнопку опубликовать, статья добавится в систему. Снизу формы расположен блок с предварительным отображением статьи, если бы она была уже опубликована.

Название статьи

Текст статьи

Категории

Картинка-превью

Выберите файл Файл не выбран

ОПУБЛИКОВАТЬ ПОСТ

Название статьи

15 Комментариев Preview/Статья 256 Лайков

Е-Mail адрес

Введите свой E-Mail

ПОДПИСАТЬСЯ

Рисунок 32. Форма добавления статьи с превью

2.2.9. Страница редактирования поста

Страница редактирования поста аналогична странице добавления нового поста, только поля формы изначально заполнены информацией статьи.

2.2.10. Страница администратора со списком статей

На данной странице расположен список всех статей с их краткой информацией, а также кнопки для перехода на страницу поста и для удаления статей из системы.

ПАНЕЛЬ АДМИНИСТРАТОРА		Посты » Авторы » Добавить автора » Выход	
Заголовок	Описание		
Как я использую AirDrop вместо Тиндера	Author: Andrei Posted at: Июнь 10, 2020, 1:21 п.п. Edited at: Июнь 10, 2020, 1:21 п.п. Categories: Информационная безопасность	Читать	Удалить
Объясните мне, как вы для себя разобрались в моделях типизаций — они же все размыты	Author: Andrei Posted at: Июнь 10, 2020, 1:27 п.п. Edited at: Июнь 10, 2020, 1:27 п.п. Categories: JavaScript Программирование .NET C# TypeScript	Читать	Удалить

Рисунок 33. Блок со списком статей на странице администратора

2.2.11. Страница администратора со списком авторов

На данной странице расположен блок со списком всех авторов с их краткой информацией, а также кнопки для сброса пароля и для удаления автора из системы.



ПАНЕЛЬ АДМИНИСТРАТОРА		Посты » Авторы » Добавить автора » Выход	
Username	E-Mail	Description	
 dendy	dendy36rus@gmail.com	First name: Andrei Last name: Borodin Last login: Июнь 10, 2020, 1:35 п.п.	Сбросить пароль Удалить
 bullovn	bullovn@yandex.ru	First name: Nikita Last name: Bullov Last login: None	Сбросить пароль Удалить

Рисунок 34. Блок со списком авторов на странице администратора

2.2.12. Страница добавления нового автора

На странице находится блок с формой для добавления нового автора, заполнив которую и нажав кнопку «Добавить», администратор добавит нового автора в систему, а на указанный E-Mail отправится письмо с данными для авторизации.

ДОБАВЛЕНИЕ НОВОГО АВТОРА

Логин нового автора

E-mail нового автора

Имя нового автора - *Optional*

Фамилия нового автора - *Optional*

ДОБАВИТЬ

Рисунок 35. Форма добавления нового автора

2.2.13. Страница панели администратора

На данной странице расположена ссылка в виде картинки для перехода в личный кабинет Яндекс.Метрики.

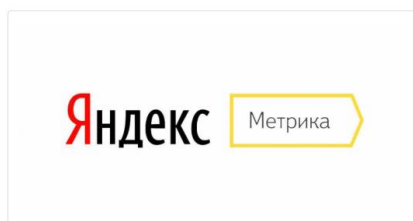


Рисунок 36. Картинка-ссылка, ведущая в личный кабинет Яндекс.Метрики

2.3. Подключение аналитики

В качестве сервиса по сбору статистики о посещениях и выполнению описанных продуктовых сценариев на сайте был выбран продукт Яндекс.Метрика.

Были описаны четыре составные цели, каждая из которых соответствует одному из продуктовых сценариев.

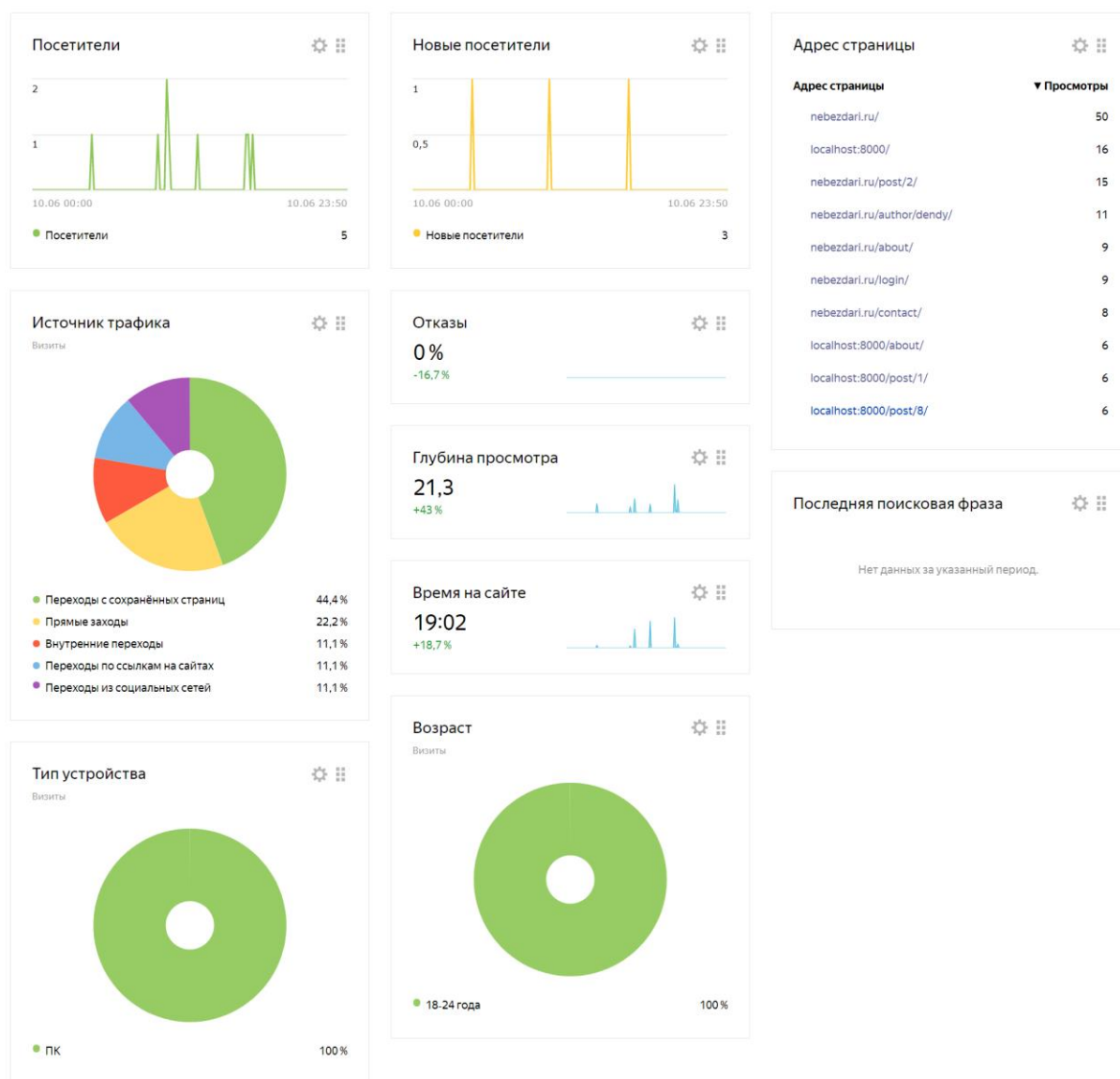


Рисунок 37. Скриншот из личного кабинета Яндекс.Метрики

Тестирование

Для тестирования приложения были использованы Unit-тесты. Была задействована стандартная библиотека тестирования Django. Так как практически все взаимодействие с приложением осуществляется через веб формы, то все формы были полностью покрыты юнит тестами. Была протестирована как обработка вводимых данных на стороне клиента, так и дальнейшая работа с уже полученными сервером данными. На вход подавался набор различных данных. В каждом наборе существовали некорректные данные для каких-то полей формы. Тестирование заключалось в том, чтобы проверить реакцию конкретных клиентских и серверных модулей на некорректные и на корректные данные. Также юнит тестами были покрыты система авторизации. Необходимо было удостовериться, чтобы при попытке зайти на недоступный для данной группы пользователей адрес, приложение корректно реагировало на это действие. В основном это реализовано с помощью редиректов, поэтому тесты проверяли, происходит ли редирект при попытке воспользоваться определенным ресурсом, если это делает уполномоченная на это группа пользователей, и, соответственно, не уполномоченная. Также были покрыты тестированием серверные модули Models и Views - основные модули приложения.

Также были проведено Usability тестирование. На конечных этапах создания приложения оно было отослано для тестирования около десятку людей разных возрастов для того, чтобы проверить насколько им удобно пользоваться. Затем собирались отзывы и на их основе вносились изменения.

Заключение

В ходе выполнения курсового проекта было реализовано работающее приложение, удовлетворяющее требованиям, которые были поставлены при составлении технического задания.

Для пользователей приложение представляет следующие возможности:

Для гостей:

1. Просмотр информации о компании, заведующей блогом
2. Просмотр статей на сайте
3. Комментирование статей
4. Отправка компании электронных сообщений через специальную форму на сайте

Для авторов, помимо возможностей гостя, система предоставляет следующий функционал:

1. Добавление и редактирование статей
2. Модерация своих статей

Для администратора представлены следующие возможности:

1. Добавление новых авторов в систему
2. Полный контроль над статьями автором и комментариями пользователей
3. Просмотр аналитической статистики Яндекс.Метрики

В процессе реализации системы, нами были выполнены следующие задачи:

- Проведён анализ предметной области с целью оценки преимуществ и недостатков аналогов разрабатываемой системы.
- Разработана модель программы с учётом проведённого анализа, в которой приводилось описание спецификации данных, связей между сущностями.
- Разработана концептуальная модель базы данных и построена логическая модель базы данных.
- На основе полученной информации с предыдущих задач, был осуществлён анализ и отбор необходимых средств разработки.
- Разработана клиентская часть приложения, отображаемая в браузере пользователя.
- Разработана серверная часть приложения, развёрнутая на удалённом хостинге.
- Проведено тестирование приложения.

Дальнейшее развитие

У нашей команды осталось несколько идей для дополнения функциональности разработанной системы, в большей мере расширяющие возможности авторов и администратора:

1. Приведение авторам подробной статистики для анализа написанных ими статей
2. Вывод статистических данных о всём сайте в панель администратора

Список использованных источников

- 1) Django documentation [Электронный ресурс]. – URL: <https://docs.djangoproject.com/en/3.0/> (дата обращения 10.06.2020)
- 2) Django REST Framework [Электронный ресурс]. – URL: <https://www.django-rest-framework.org/> (дата обращения 10.06.2020)
- 3) Django Swagger – Yet Another Swagger Generator [Электронный ресурс]. – URL: <https://drf-yasg.readthedocs.io/en/stable/> (дата обращения 10.06.2020)
- 4) Введение в Django. Что такое Django [Электронный ресурс]. – URL: <https://metanit.com/python/django/1.1.php> (дата обращения 10.06.2020)
- 5) Леоненков А.В. Самоучитель UML. – 2-е изд. / А.В. Леоненков – СПб.: БХВ-Перербург, 2004. – 432 с.: ил.

Приложения

ПРИЛОЖЕНИЕ А

Диаграмма IDEF0

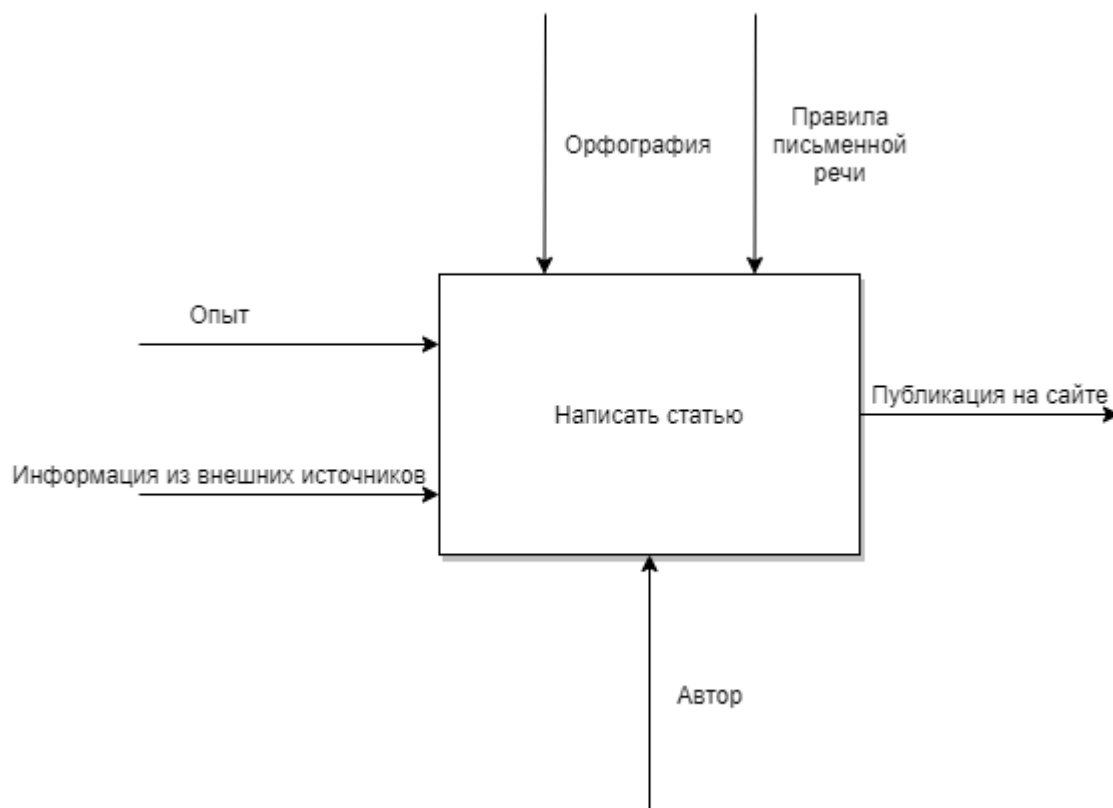


Диаграмма вариантов использования. Администратор.

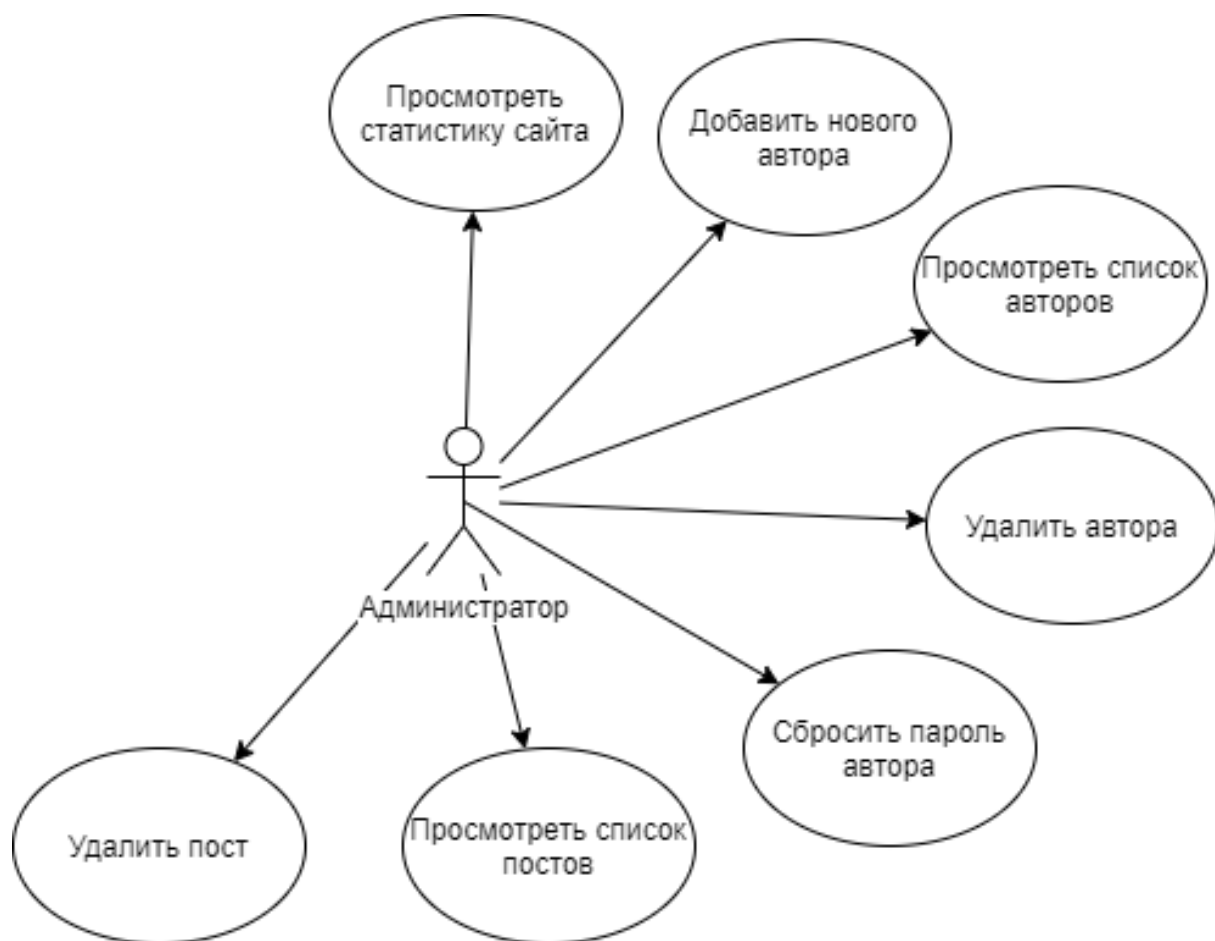
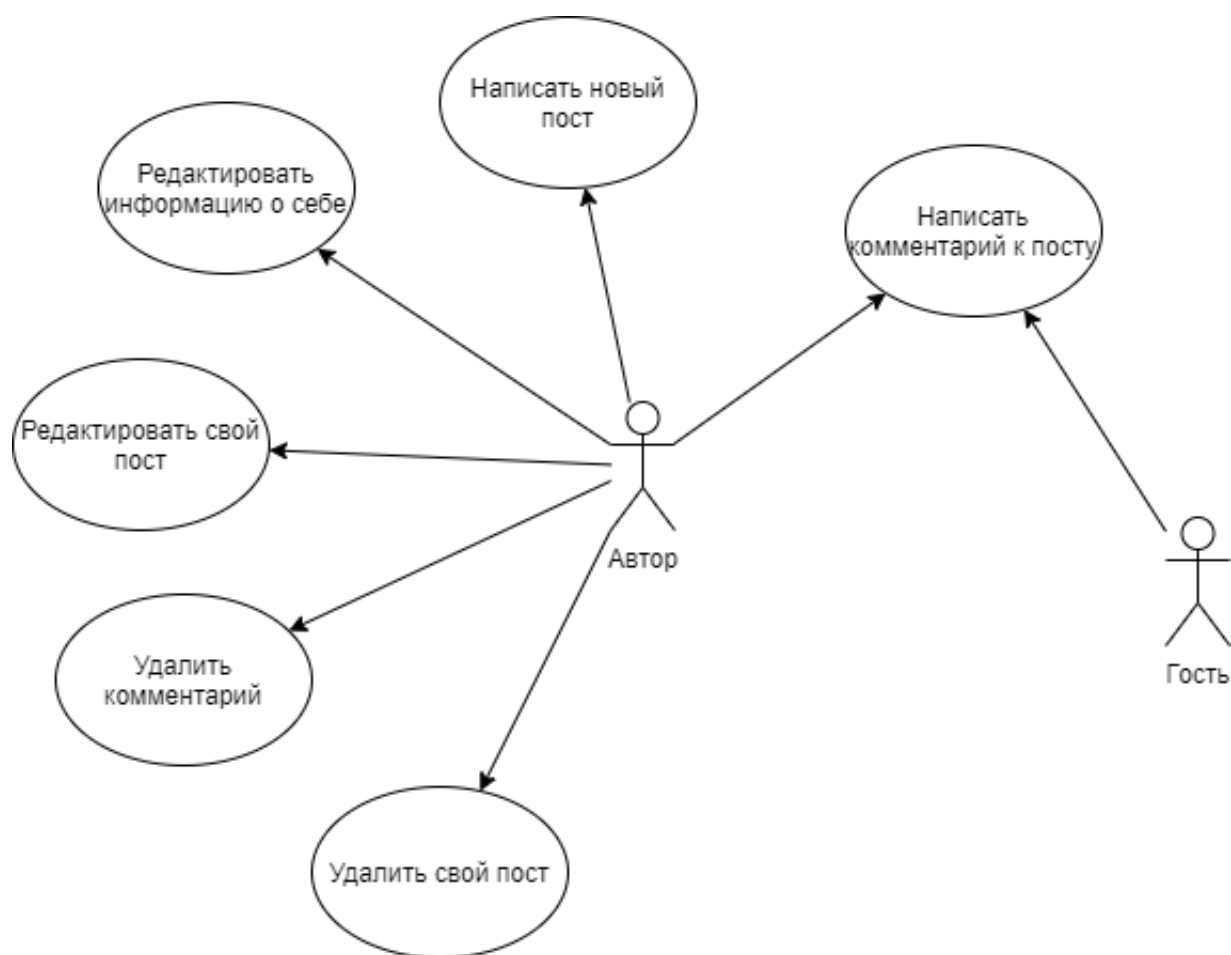


Диаграмма вариантов использования. Автор и гость.



**Диаграмма вариантов использования. Авторизация
пользователей.**

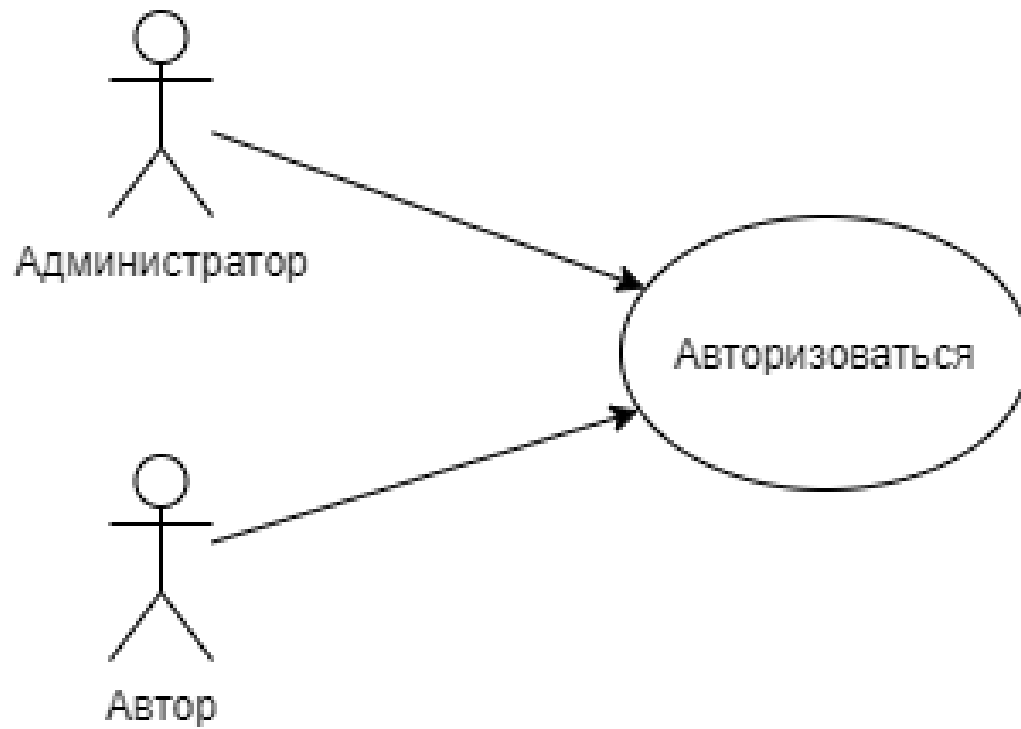


Диаграмма вариантов использования. Общие возможности пользователей.

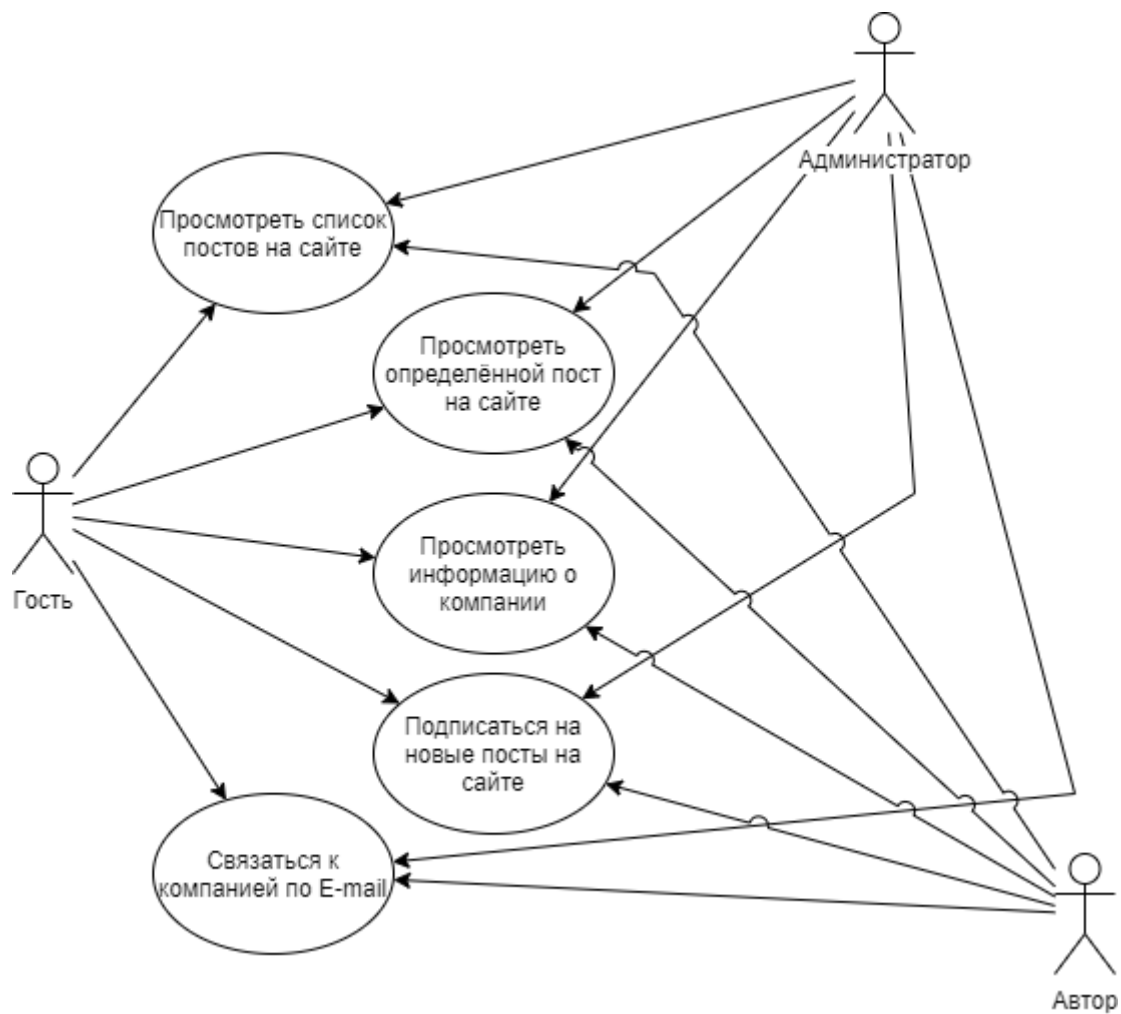


Диаграмма классов

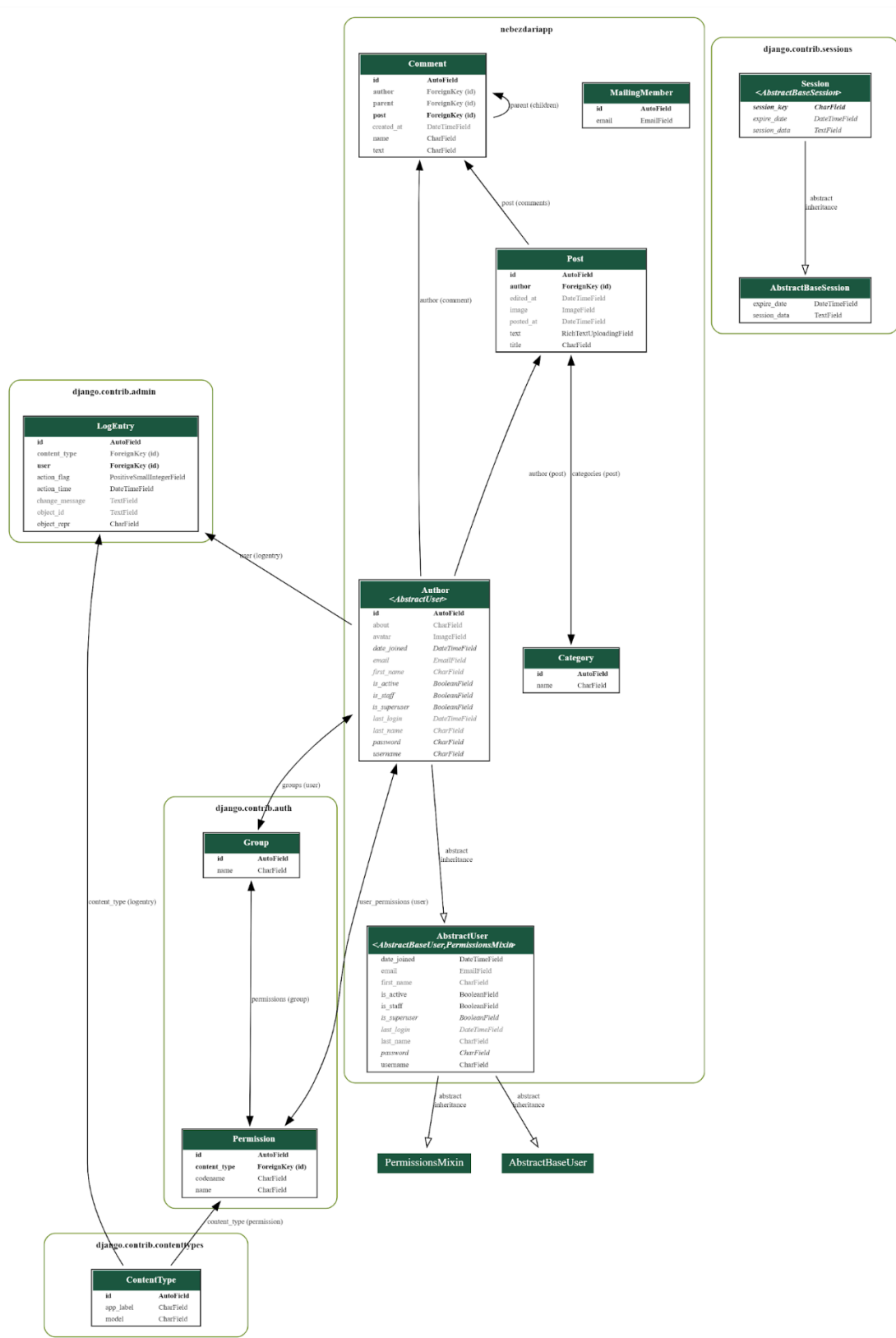
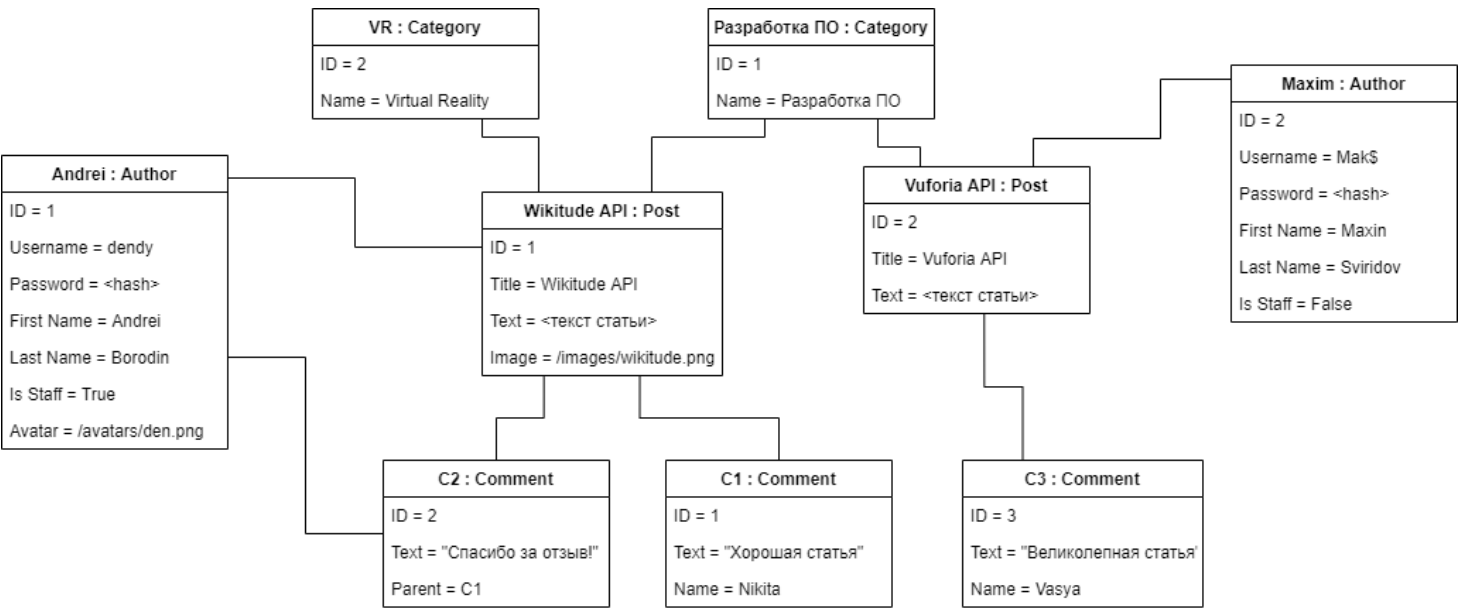


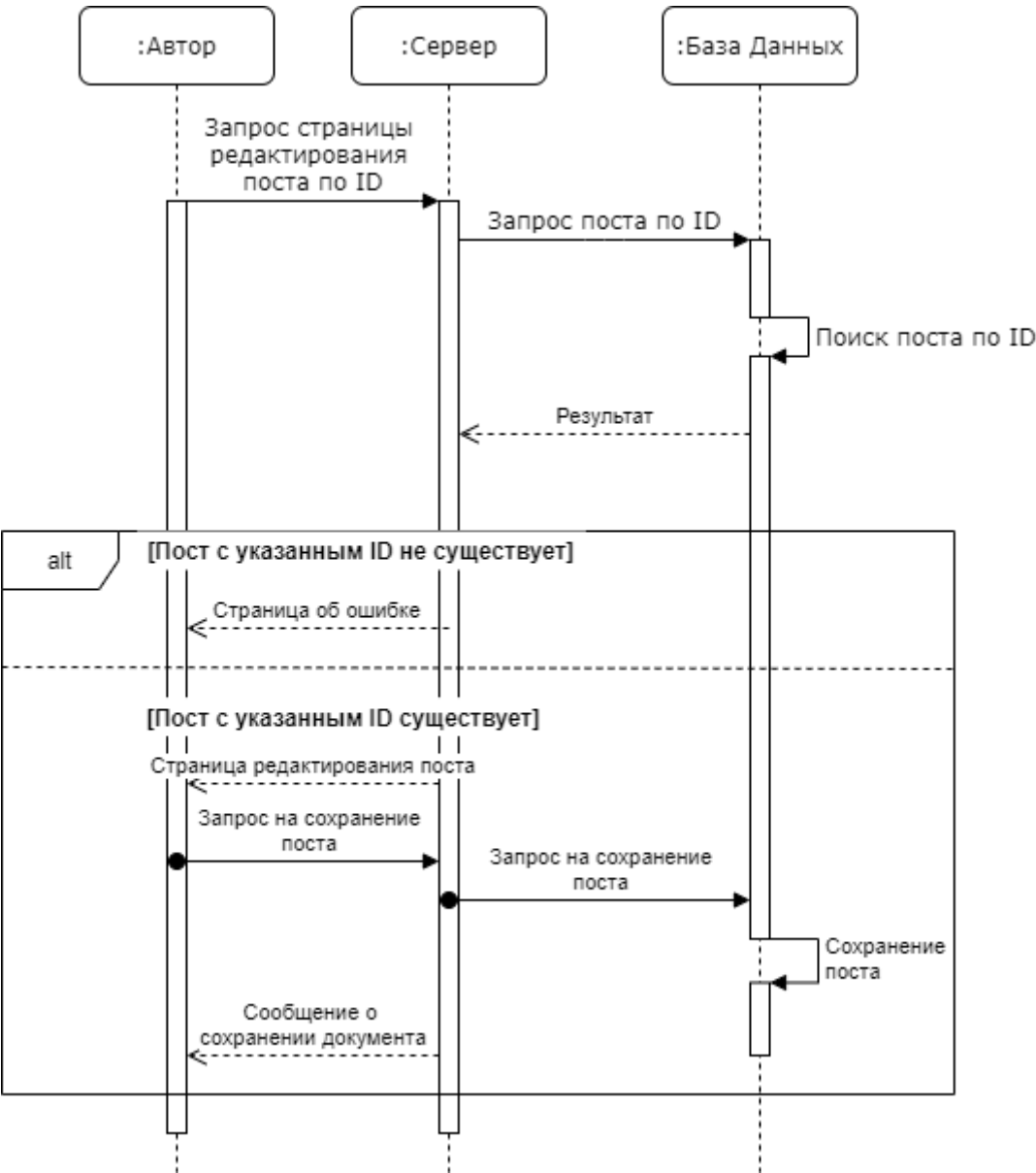
Диаграмма объектов



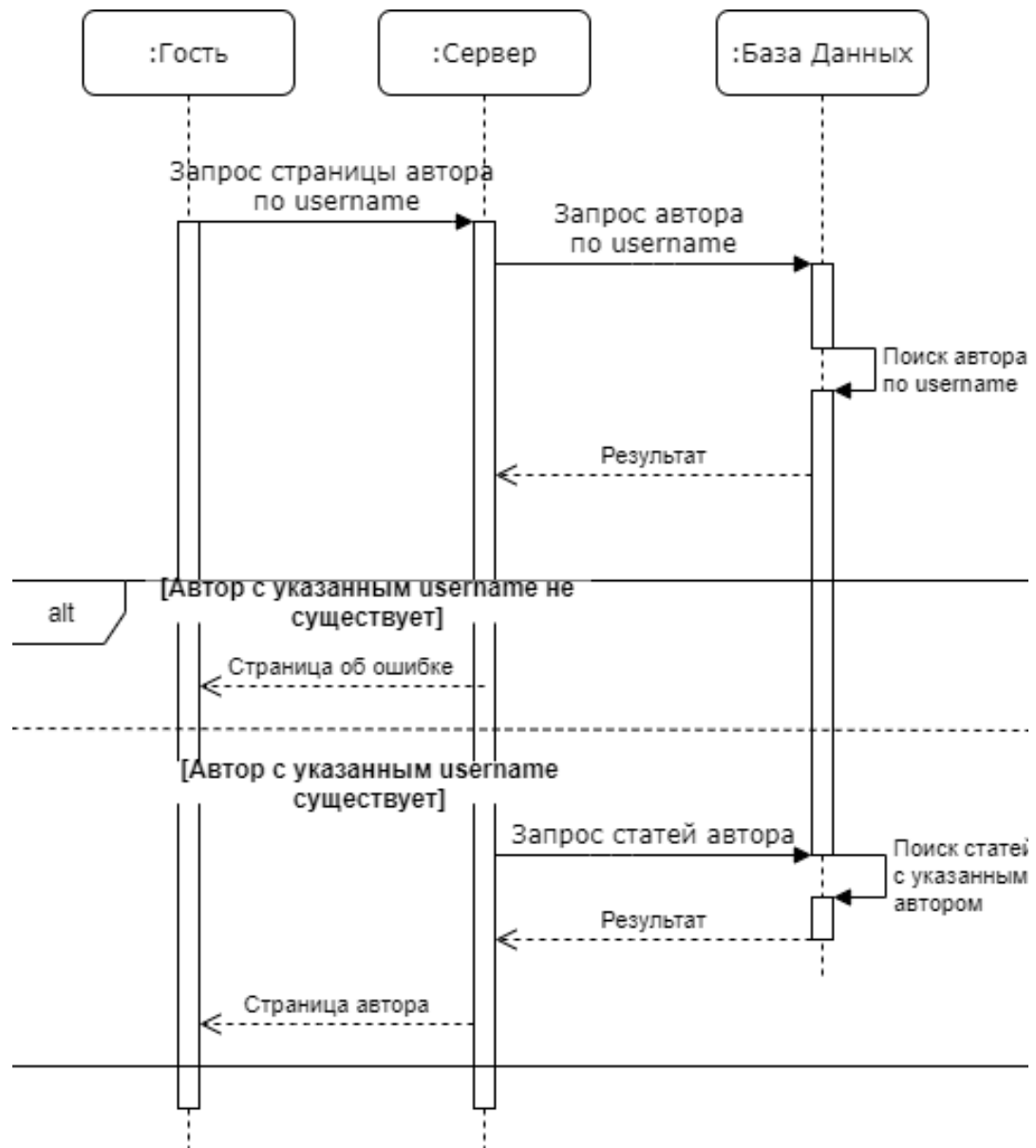
**Диаграмма взаимодействия. Добавление администратором
нового автора.**



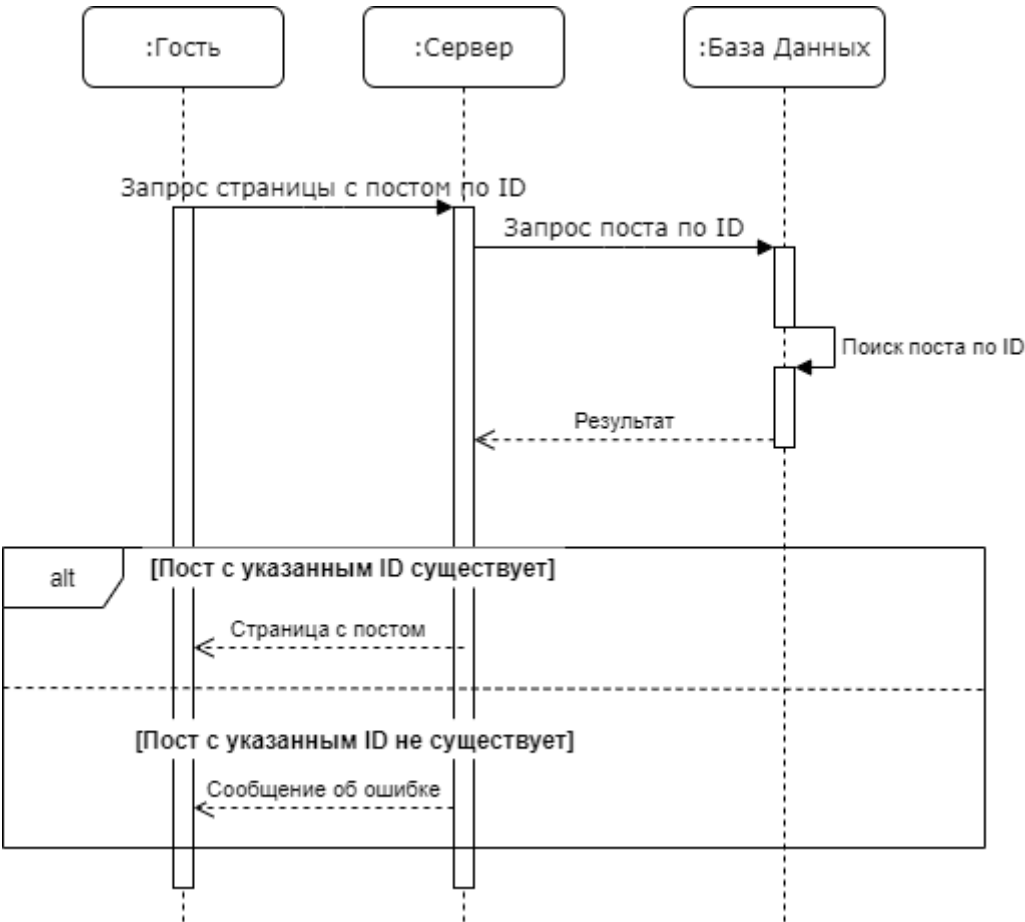
Диаграмма последовательности. Редактирование автором статьи.



**Диаграмма последовательности. Просмотр пользователем
страницы автора.**



**Диаграмма последовательности. Просмотр пользователем
страницы с содержанием статьи.**



ПРИЛОЖЕНИЕ Ж

Диаграмма активностей. Авторизация пользователя в системе.

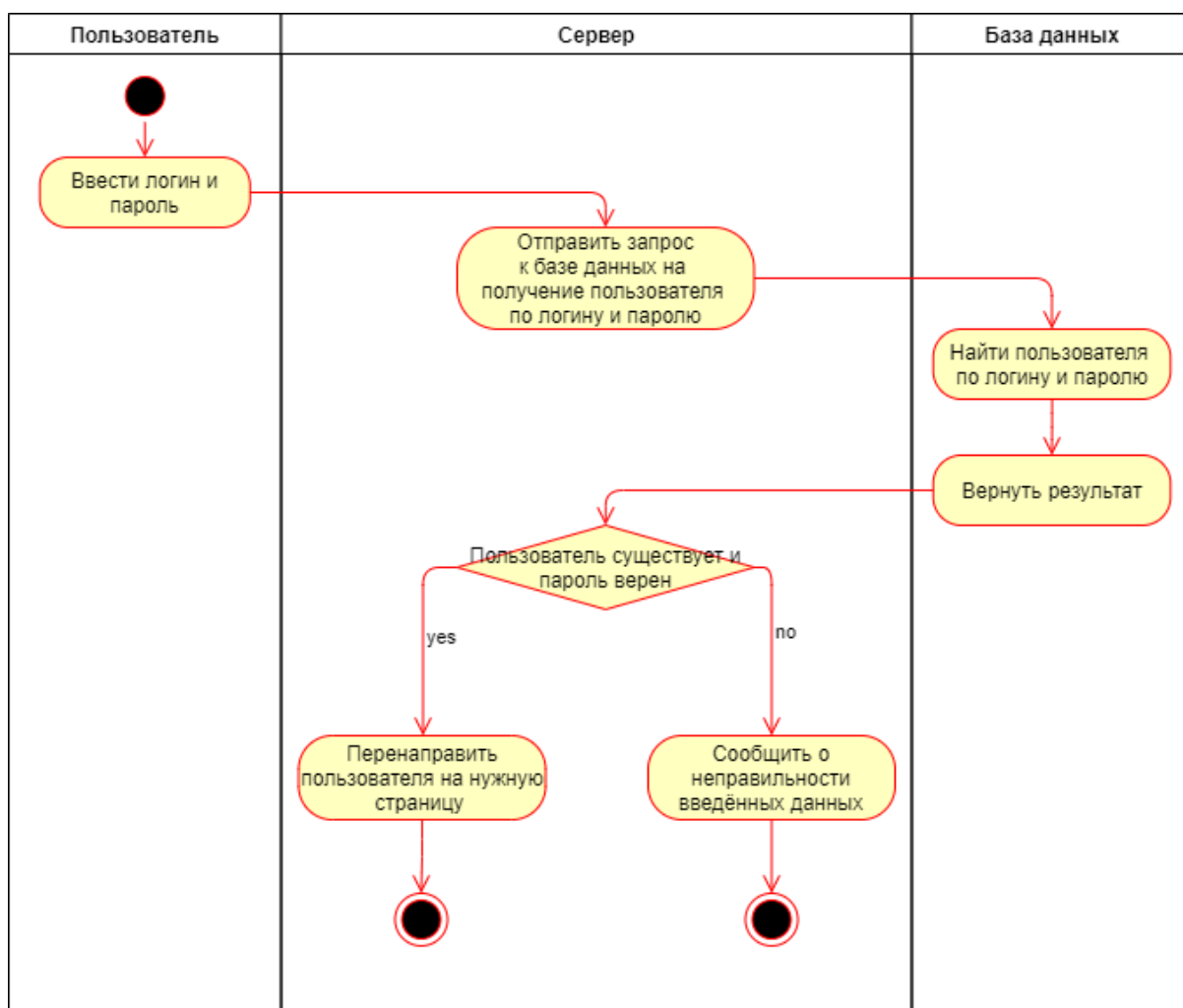


Диаграмма состояний. Взаимодействие пользователя с системой.

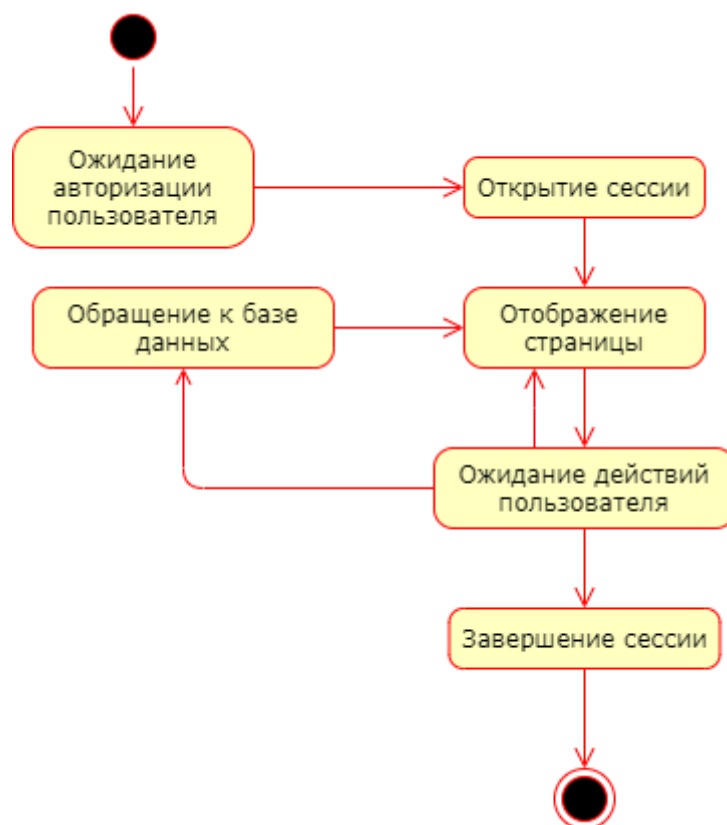
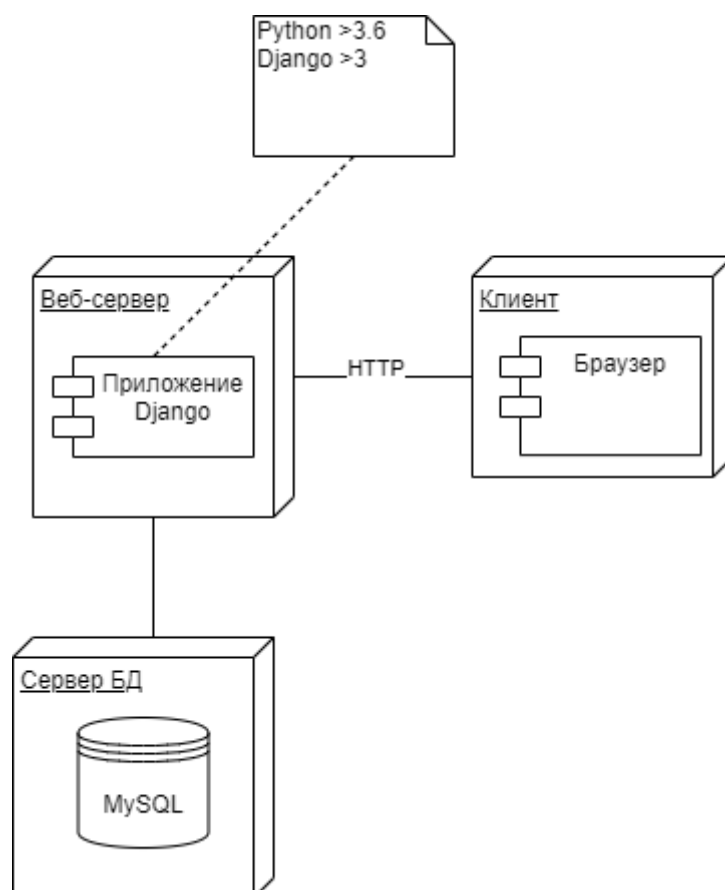


Диаграмма развертывания



Скриншот воронок аналитики и Яндекс Метрики.

Название цели		Описание	Номер цели
1	Просмотр гостем статьи	<div> <div>составная цель</div> <div> Переход на главную страницу (ID: 99337330): <ul style="list-style-type: none"> url: содержит «nebezdari.ru» </div> <div> Нажатие на кнопку фильтрации постов (ID: 99337333): <ul style="list-style-type: none"> событие: идентификатор цели «filter» </div> <div> Переход на страницу статьи (ID: 99337336): <ul style="list-style-type: none"> url: начинается с «nebezdari.ru/post/» </div> </div>	99337300
2	Подписка гостей на рассылку	<div> <div>составная цель</div> <div> Переход на главную страницу (ID: 99337561): <ul style="list-style-type: none"> url: совпадает «nebezdari.ru» url: совпадает «nebezdari.ru/» </div> <div> Переход на страницу поста (ID: 99337564): <ul style="list-style-type: none"> url: содержит «nebezdari.ru/post/» </div> <div> Нажать кнопку "Подписаться" (ID: 99337570): <ul style="list-style-type: none"> событие: идентификатор цели «subscribe_button» </div> </div>	99337558
3	Авторизация авторов	<div> <div>составная цель</div> <div> Переход на главную страницу (ID: 99338605): <ul style="list-style-type: none"> url: совпадает «nebezdari.ru» url: совпадает «nebezdari.ru/» </div> <div> Переход на страницу авторизации (ID: 99338608): <ul style="list-style-type: none"> url: содержит «nebezdari.ru/login/» url: содержит «nebezdari.ru/author/login» </div> <div> Нажать кнопку "Авторизоваться" (ID: 99338614): <ul style="list-style-type: none"> событие: идентификатор цели «login_button» </div> <div> Переход на страницу автора (ID: 99822319): <ul style="list-style-type: none"> url: начинается с «nebezdari.ru/author/» </div> </div>	99338365
4	Добавление автором новой статьи	<div> <div>составная цель</div> <div> Переход на страницу добавления статьи (ID: 99338803): <ul style="list-style-type: none"> url: содержит «nebezdari.ru/post/add» </div> <div> Нажать кнопку "Опубликовать" (ID: 99338809): <ul style="list-style-type: none"> событие: идентификатор цели «post_button» </div> </div>	99338800