ПЕРЕЧЕНЬ УСЛОВНЫХ ОБОЗНАЧЕНИЙ, СИМВОЛОВ И ТЕРМИНОВ

В данной пояснительной записки применяются следующие сокращения и определения:

API (Application Programing Interface) – интерфейс программирования приложений

Email (electronic mail) - технология и служба по пересылке и получению электронных сообщений (электронная почта)

REST (Representational State Transfer) - архитектурный стиль взаимодействия компонентов распределенного приложения в сети.

СУБД (Система управления базами данных) - совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

SQL (structured query language) - декларативный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей системой управления базами данных.

CRUD (create read update delete) - акроним, обозначающий четыре базовые функции, используемые при работе с базами данных.

HTTP (HyperText Transfer Protocol) - протокол прикладного уровня передачи данных.

HTTPS (HyperText Transfer Protocol Secure) - расширение протокола HTTP для поддержки шифрования в целях повышения безопасности.

JSON (JavaScript Object Notation) - текстовый формат обмена данными, основанный на JavaScript.

URL (Uniform Resource Locator) - система унифицированных адресов электронных ресурсов, или единообразный определитель местонахождения ресурса.

ACID (atomicity, consistency, isolation, durability) - представляет собой набор свойств транзакций базы данных, предназначенный для гарантии достоверности данных, несмотря на ошибки, сбои питания и другие неполадки.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

С каждым годом все больше людей предпочитают удаленную работу. Хотя фриланс у многих людей вызывает противоречивые чувства. Для одних — это сидеть на красивом пляже с ноутбуком на коленях и бокалом чего-нибудь в одной руке, для других — нестабильность и отсутствие фиксированного заработка. На данный момент он наиболее популярен в таких сферах, как дизайн, копирайтинг, верстка сайтов, создание мобильных приложений и так далее. Основными минусами такого рода занятий являются:

1. Отсутствие соцпакета. У данной группы людей отсутствуют больничные, пенсионное обеспечение, а также многие льготы, которые предоставляют компании своим сотрудникам.
2. Нестабильность заработка. Это особенно чувствуется начинающим фрилансерам, у которых нет постоянной клиентской базы. Также прибыль сильно зависит не только от самого фрилансера, но и от окружающей ситуации в целом.
3. Необходимость самому планировать и организовывать свое рабочее место и время.
4. Иногда возникает недостаток коммуникации

Несмотря, на существенный список минусов, положительных моментов у такой работы тоже достаточно. Например:

1. Возможность самому выбирать удобные часы работы.
2. Нет необходимости поездок в офис. Это не только позволяет экономить время, но так же и деньги на проезд, а еще не загрязнять лишний раз окружающую среду тем людям, которые привыкли ездить в офис на автомобиле.
3. Возможность создать комфортные условия для работы. Каждый человек представляет удобное рабочее место по-разному: для одних это полная тишина, а другие предпочитают фоновые звуки, кто-то работает сидя, а кто-то стоя и так далее.
4. Возможность выбирать интересные задания.

Таким образом, фриланс достаточно популярный вид заработка, а значит, что и сайт для поиска таких услуг будет иметь достаточно много клиентов.

1. АНАЛИЗ АНАЛОГОВ И ФОРМИРОВАНИЕ ТРЕБОВАНИЙ К ПРОГРАММНОМУ СРЕДСТВУ

1.1. Обзор существующих аналогов

В качестве аналогов было решено проанализировать 3 крупнейших сайта Республики Беларусь по оказанию фриланс услуг: freelancehunt, kabanchik, фрилансер.бел.

1.1.1. Обзор сайта freelancehunt

Главная страница сайта представлена на рисунке 1.1.

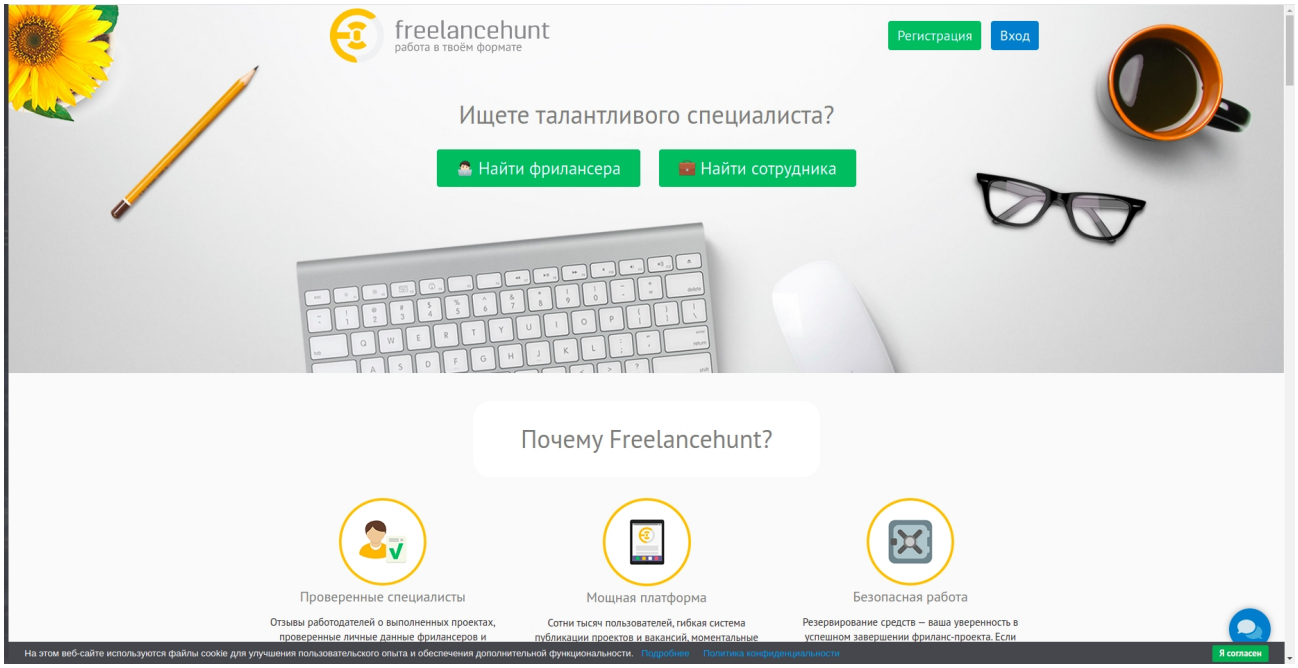

Рисунок 1.1 – сайт «Freelancehunt»

Данный сайт является достаточно популярным, им пользуются больше 200000 фрилансеров из Республики Беларусь . Основные его достоинства:

- Возможность уже при регистрации выбрать, ты ищешь работу или предлагаешь ее. Этот функционал позволяет проще ориентироваться на сайте, а также позволяет использовать различный функционал для разных типов пользователей
- Возможность поставить рейтинг в комментарии. Это позволяет увидеть относительную оценку работу, что часто более информативно, чем текст, а также избавить от необходимости писать текст комментария в целом.

- Наличие рейтинга у каждой конкретной услуги. Данный рейтинг высчитывается как среднее арифметическое всех рейтингов из комментариев к данной услугу и является важным для заказчика фактором, при выборе того или иного работника.

Основные недостатки сайта:

- Для регистрации необходим код, полученный по смс, которые достаточно долго не приходят(часто время ожидания больше часа). Невозможность относительно быстро и просто зарегистрироваться с большой вероятностью может стать причиной того, что пользователь предпочтет другой сайт.
- Отсутствие персональной страницы у каждой услуги. Это не позволяет должным образом узнать все ее подробности, а также увидеть комментарии, которые оставили люди, заказавшие ее.
- Возможность заказа только услуги, не требующей специализированного оборудования, кроме персонального компьютера. Это существенно сужает спектр все оказываемых услуг, что скорее всего станет причиной выбрать другой сайт как для работников, так и для заказчиков.
- Для создания заказа на услугу необходима обязательная регистрация. Данная необходимость может стать проблемой для тех заказчиков, которые хотят единоразово или редко пользоваться услугами сайта, что повлечет за собой отток пользователей.

1.1.2. Обзор сайта kabanchik.by

Главная страница сайта представлена на рисунке 1.2.

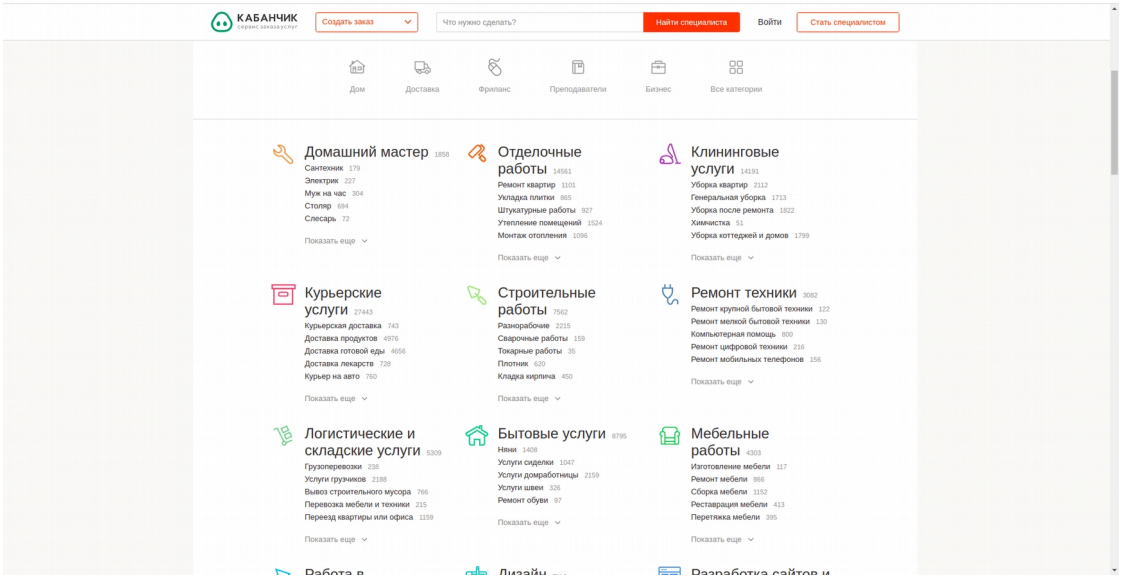

Рисунок 1.2 – сайт «Kabanchik»

Данный сайт привлекает клиентов широким спектром оказываемых услуг, что видно на рисунке 1.2. Также основными достоинствами сайта являются:

- Возможность заказать услугу с выездом к вам домой. Это позволяет значительно расширить спектр оказываемых услуг, а также становится значительным достоинством для людей, которые боятся еще больше повредить сломанный объект при перевозке либо если он слишком массивный.
- Возможность заказать услугу без регистрации. Это привлекает людей, которые хотят единоразово воспользоваться услугами сайта, а также людей, которые хотят заказать услуг в кратчайшие сроки.
- Широкий спектр оказываемых услуг.Это очень важное достоинство, тк большинство людей предпочтут пользоваться одним сайтом, где можно заказать большинство необходимых услуг, чем различными сайтами для заказа каждой из них по отдельности.

Основные недостатки:

- Невозможность работником установить минимальную цену. Из-за отсутствия такой возможности клиентам, которые заказывают услугу в первый раз, будет сложно предложить цену. Так же это является большим недостатком и для работника, тк он будет получать большое количество предложений, которые ему не подойдут из-за предложенной оплаты.
- Невозможность добавить фотографии к комментарию. Очень часто фотография может избавить от написания текстового комментария, а также более явно продемонстрирует все достоинства и недостатки выполненной работы.
- Нет пояснений о том, работа выполняется у заказчика на дому или необходимо привести сломанную вещь исполнителю. Об этом заказчику с исполнителем придется договариваться лично, и как следствие часть заказов отменяется из-за несовпадения возможностей.

1.1.3. Обзор сайта фрилансер.бел

Главная страница сайта представлена на рисунке 1.3.

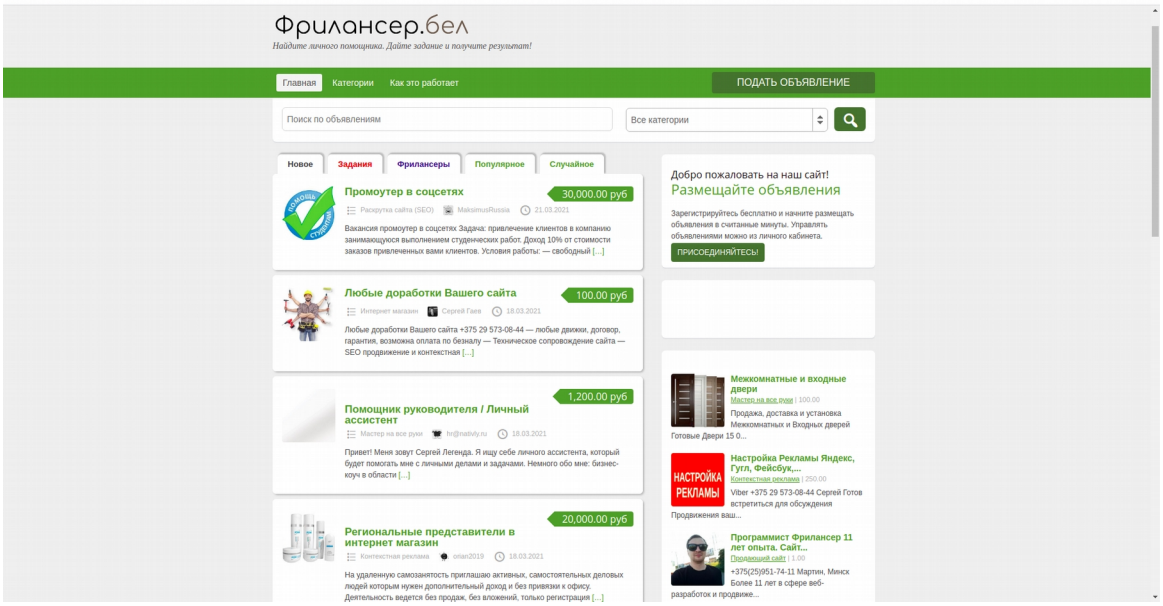

Рисунок 1.3 – сайт «фрилансер.бел»

На главной странице данного сайта отображен список доступных услуг, что облегчает заказчикам их поиск. Также достоинствами сайта являются:

- На персональной странице каждой услуги написаны комментарии, оставленные к ней. Это позволяет увидеть, что заказчики думают о предоставлении конкретно этой услуги, а не исполнителе в целом, что позволит лучше выбрать услугу.
- На персональной странице каждой услуги дано ее развернутое описание. Это позволяет работникам оставить какие-либо важные факты и комментарии о выполнении этой работы.

Основные недостатки:

- Необходимость регистрации, для заказа услуги. Как уже говорилось выше, это может вызвать большой отток пользователей сайта.
- Нет возможности для работника установить различные минимальные цены для случая покупки им материалов либо же материалов заказчика. Это является существенным минусом, так как цены на выполнение заказа с наличием материала и его отсутствием могут сильно различаться и как следствие, заказчик не будет знать

реально необходимую сумму. Это так же не дает заказчику выбрать более выгодный для него вариант.

- Нет возможности подать жалобу. Из-за невозможности подать жалобу, недобросовестные заказчики и работники будут продолжать пользоваться сайтом, что приведет не только к потере денежных средств большое количество людей, но также и сильно подорвет доверие к этому сайту, тем самым уменьшив его популярность.
- Нет рейтингов у конкретных услуг. Это не позволяет пользователем быстро узнать, насколько хорошо работник справляется с конкретно этой задачей, и возникает необходимость чтения всех комментариев, если они существуют.

1.2. Вывод по результатам обзора аналогов программного средства

В ходе анализа аналогов было решено, что back-end для сайта должен выполнять следующие функции:

- Регистрация.
  Однако, она должна быть необязательной для заказа услуги. Это необходимо, чтобы привлечь тех пользователей, которые собираются редко что-либо заказывать. При регистрации так же должна быть возможность выбрать, кем вы желаете быть: заказчиком или работником. Это позволит отображать только нужную человеку информацию и сделает сайт более простым в использовании. Так же она должна проводиться по email и паролю, однако с возможностью добавления фотографии и мобильного телефона.
- Вход.
- Возможность создать услугу.
  Эта возможность должна быть доступна только мастерам. Также должна быть возможность указать выполняется услуга дома у заказчика или работника, будут использоваться материалы заказчика или работника и соответственно различные минимальные цены. Работник может изменить или удалить любую из своих услуг, а все заказчики, которые ее заказали должны быть уведомлены об этом по email, в случае, если они его указали.
- Возможность создать заявку на услугу.
  Эта возможность должна предоставляться без обязательной регистрации, однако необходимо в таком случае ввести или email,

или номер телефона, или оба сразу. Необходимо также указать желаемое начало выполнения заказ. Заявку невозможно создать пользователю, который занесен в черный список, а также по email, адресу или телефонному номеру, которые находятся в нем. Нельзя также создать заявку на прошедшее время, или если на это время или тридцатиминутный промежуток в обе стороны уже есть другие, допущенные работником заявки. Свой неодобренный заказ на услугу заказчик может изменить или удалить, однако, если он был незарегистрирован, ему придется так же еще раз указать свой email или телефон(смотря, что он указывал при заказе), чтобы подтвердить, что заказ именно его.

- Возможность ответить на эту заявку.
  Эта возможность предоставляется пользователям-работникам, и пользователь, на чью заявку ответили должен быть извещен об этом по почте, в случае, если он указал ее.
- Возможность оставить комментарий и оценить заказанную услугу.
  Эта возможность должна предоставляться только зарегистрированным пользователям, которые оставляли заказ на эту услугу и ее одобрил мастер. Обязательно оставить или текст, или оценку по пятибалльной шкале, или и то, и другое. Должна быть возможность добавить фотографии. При этом рейтинг услуги, на которую был оставлен комментарий с оценкой, должен составлять среднее арифметическое всех оставленных оценок. Пользователь может удалить или изменить свой комментарий, при этом рейтинг соответствующей услуги должен быть пересчитан.
- Возможность оставить жалобу.
  Любой зарегистрированный пользователь может оставить жалобу, для этого ему необходимо указать email, или телефон или оба сразу того, на кого он подает жалобу. Можно также указать адрес и добавить фотографии и документы. Все администраторы сайта должны быть извещены по e mail о появлении новой жалобы, которую необходимо рассмотреть
- Возможность администраторам ответить на жалобу.
  Администратор может ответить на жалобу, при этом подавший ее пользователь должен быть об этом извещен по email.
- Возможность добавить пользователя в черный список.
  Администратор может добавить в черный список адрес, email, или телефон, при этом необходимо указать либо e mail, либо телефон, либо и то, и другое. Это сделано для того, чтобы заказ не могли осуществить даже незарегистрированные пользователи, которые нарушали правила(их телефон или email будет в черном списке и они не смогут ничего заказать). Если при добавлении был указан email, по нему должно быть выслано извещение о том, что он

добавлен в черный список. Администратор также может удалить запись из черного списка, в случае присутствия там email, по нему так же должно быть отправлено письмо об удалении из черного списка.

- Возможность редактировать пользователя. Зарегистрированный пользователь может сам себе поменять телефон, имя, фамилию, фото, и статус мастера(определяет, пользователь является работником или заказчиком). Администратор может также поменять у пользователя статус мастера и сделать пользователя администратором или наоборот. Администратор также может пометить пользователя, как удаленного, такой пользователь не сможет больше зайти на сайт. Администратор не может изменить информацию о себе или удалить себя
- Возможность просмотреть информацию. Пользователям, в зависимости от их роли должна быть предоставлена возможность просматривать необходимую информацию.

1.3. Формирование требований

В рамках дипломного проекта поставлена цель разработать back end для сайта по оказанию фриланс услуг с использованием фреймворка Django. В соответствии с проанализированными аналогами к проекту будут предъявляться следующие функциональные требования:

1. Любой пользователь должен иметь возможность просмотреть информацию о услугах, которые можно заказать, с возможностью поиска по имени. Для этого необходимо отобразить следующие поля:
    - Название услуги.
    - Ее описание.
    - Оказывается услуга с выездом или нет.
    - Необходимо ли предоставить материалы.
    - Минимальная стоимость с учетом покупки материалов.
    - Минимальная стоимость без учета покупки материалов.
    - Информация о рабочем:
        - Емэйл.
        - Телефон.
        - Имя.
        - Фамилия.

2. Любой пользователь должен иметь возможность перейти на страницу услуги. На ней, должны быть отображены следующие поля:
    - Название услуги.
    - Ее описание.
    - Оказывается услуга с выездом или нет.
    - Минимальная стоимость с учетом покупки материалов.
    - Минимальная стоимость без учета покупки материалов.
    - Рейтинг.
    - Необходимо ли предоставить материалы.
    - Комментарии заказчиков:
        - Выставленный рейтинг.
        - Текстовый коммент.
    - Информация о рабочем:
        - Е мэйл.
        - Телефон.
        - Имя.
        - Фамилия.
3. Пользователь должен иметь возможность зарегистрироваться при помощи формы. Форма регистрации пользователя-клиента должна содержать следующие поля:
    - Электронная почта(максимальное количество символов 150).
    - Пароль(максимальное количество символов 120).
    - Подтверждение пароля(максимальное количество символов 120).
    - Номер телефона (не обязательно к заполнению).
    - Имя(максимальное количество символов 150).
    - Фамилия(максимальное количество символов 150).
    - Фотография (не обязательно к заполнению).
    - Является данный пользователем мастером, предлагающим услугу, или заказчиком (не обязательно к заполнению, по умолчанию при регистрации любой пользователь является мастером).
4. Пользователь должен иметь возможность авторизоваться при помощи формы. Форма должна содержать следующие поля:

- Электронная почта(максимальное количество символов 150).
- Пароль(максимальное количество символов 120).

Все поля обязательны для заполнения. Далее приложение должно проверить правильность введенных данных. Если пользователь заблокирован, то ему выводится сообщение о блокировке.

5. Зарегистрированному пользователю должна быть предоставлена возможность сменить :

- номер телефона.
- имя(максимальное количество символов 150).
- фамилию(максимальное количество символов 150).
- является ли данный пользователь мастером.

Все поля не обязательны для заполнения. Приложение должно произвести проверку корректности введенных данных (соответствие нового пароля требованиям, новый пароль не должен совпадать со старым, пользователь может обновить информацию только о самом себе). Если введенные данные некорректны, то пользователю должно быть выведено сообщение об этом.

6. Пользователю-администратору должна быть предоставлена возможность удалить пользователя (при этом должна быть произведена проверка, что пользователь не удаляет сам себя). Удаленный пользователь должен помечаться соответствующим статусом в базе данных и не иметь возможности залогиниться. Ему так же на емэйл должно быть отправлено письмо о том, что его удалили с сайта.
7. Пользователь-администратор также может присвоить пользователю статус администратора или мастера, или наоборот убрать их (при этом должна быть произведена проверка, что пользователь не меняет статус сам себе). При условии успешной смены статуса, пользователю, которого изменили, должен быть отправлен email соответствующего содержания.
8. Пользователю-администратору должна быть доступна возможность получения списка всех зарегистрированных пользователей, со следующими полями:
    - email.
    - телефонный номер.
    - имя.
    - фамилия.

    Данный список должен иметь возможность фильтрации по:
    - email.

- статусу администратора.
- статусу мастера.
- статусу удаления.

9. Пользователю-администратору должна быть доступна возможность просмотра информации о конкретном пользователе, должны быть отображены следующие поля:
    - email.
    - телефонный номер.
    - имя.
    - фамилия.
    - фотография.
    - статус администратора.
    - статус мастера.
    - статус удаления.
10. Зарегистрированному пользователю-мастеру должна быть предоставлена возможность добавить оказываемую услугу. Для этого необходимо предоставить следующую информацию:

- Название услуги (выбирается из списка, максимальное количество символов 50).
- Возможность выполнения работы на дому заказчика(не обязательно к заполнению, принимает следующие значения: нет, да. По умолчанию - да).
- Текстовое описание (не обязательно к заполнению, максимальное количество символов 150).
- Необходимость предоставления материалов (может принимать следующие значения: да, нет, оба варианта).
- Минимальная цена без учета покупки материалов (не обязательно к заполнению, минимальное значение 0.01, максимальное значение 10000).
- Минимальная цена с учетом покупки материалов (не обязательно к заполнению, минимальное значение 0.01, максимальное значение 10000).

  При создании услуги необходимо произвести следующие проверки:
    - Если наличие материалов необходимо, то должна быть указана минимальная цена без учета покупки материалов и отсутствовать минимальная цена с учетом покупки материалов.

- Если наличие материалов не нужно, то должна быть указана минимальная цена с учетом покупки материалов и отсутствовать минимальная цена без учета покупки материалов.
- Если возможны оба варианта, то должна быть указана и минимальная цена с учетом покупки материалов, и минимальная цена без учета покупки материалов.

11. Зарегистрированному пользователю-мастеру должна быть предоставлена возможность просмотреть список всех его услуг с возможностью поиска по названию. Для каждой конкретной услуги должны быть отображены следующие поля:
    - Название услуги.
    - Ее описание.
    - Выезд на дом к заказчику.
    - Необходимо ли предоставить материалы.
    - Минимальная стоимость с учетом покупки материалов.
    - Минимальная стоимость без учета покупки материалов.
    - Информация о рабочем:
        - Емэйл.
        - Телефон.
        - Имя.
        - Фамилия.
12. Зарегистрированному пользователю-мастеру должна быть предоставлена возможность обновить выставленную им услугу. Для этого необходимо предоставить следующие поля:

- Выполнение работы на дому заказчика(не обязательно к заполнению).
- Текстовое описание (не обязательно к заполнению, максимальное количество символов 150).
- Необходимость наличия материалов (не обязательно к заполнению, может принимать следующие значения: да, нет, оба варианта).
- Минимальная цена без учета покупки материалов (не обязательно к заполнению, минимальное значение 0.01, максимальное значение 10000).
- Минимальная цена с учетом покупки материалов (не обязательно к заполнению, минимальное значение 0.01, максимальное значение 10000).

При этом приложение должно произвести следующие проверки:

- Если наличие материалов необходимо, то должна быть указана минимальная цена без учета покупки материалов и отсутствовать минимальная цена с учетом покупки материалов.
- Если наличие материалов не нужно, то должна быть указана минимальная цена с учетом покупки материалов и отсутствовать минимальная цена без учета покупки материалов.
- Если возможны оба варианта, то должна быть указана и минимальная цена с учетом покупки материалов, и минимальная цена без учета покупки материалов

В случае успешного обновления услуги, всем пользователям, которые имеет активные и подтвержденные заявки на ее выполнение должен быть отправлен email о том, что услуга была изменена.

13. Зарегистрированному пользователю-мастеру должна быть предоставлена возможность удалить выставленную им услугу. При этом услуга должна сохраниться в базе, но быть недоступна для просмотра и заказа. Всем пользователям, которые имеет активные и подтвержденные заявки на ее выполнение должен быть отправлен email о том, что услуга была удалена.
14. Зарегистрированному пользователю должна быть предоставлена возможность написания жалобы на другого пользователя. Для этого необходимо заполнить следующие поля:
    - Email пользователя, на которого подается жалоба(не обязательно для заполнения, максимальное количество символов 150).
    - Телефон пользователя, на которого подается жалоба(не обязательно для заполнения).
    - Адрес пользователя, на которого подается жалоба(не обязательно для заполнения, максимальное количество символов 100).
    - Текстовый комментарий(не обязательно для заполнения, максимальное количество символов 500).
    - Фотографии(не больше 5, не обязательно для заполнения).
    - Документы(не больше 5, не обязательно для заполнения).

    При этом приложение должно проверить наличие либо телефона,

либо e mail пользователя, на которого подается жалоба. В случае успешного создания жалобы, ей присваивается статус активной и всем пользователям-администраторам должен быть отправлен email о том, что создана новая жалоба, которую необходимо рассмотреть.

15. Пользователю-администратору должна быть предоставлена возможность просмотра всех жалоб с возможностью их фильтрации по статусу. Для этого должны быть отображены следующие поля:
    - Статус жалобы(активна, удовлетворена или отклонена).
    - Дата создания.
    - Текст.
16. Пользователю-администратору должна быть предоставлена возможность просмотра конкретной жалобы. Для этого должны быть отображены следующие поля:
    - Статус жалобы(активна, удовлетворена или отклонена).
    - Дата создания.
    - Email пользователя, на которого подается жалоба.
    - Телефон пользователя, на которого подается жалоба.
    - Адрес пользователя, на которого подается жалоба.
    - Текстовый комментарий.
    - Фотографии.
    - Документы.
17. Каждому зарегистрированному пользователю должна быть предоставлена возможность просмотра всех жалоб, созданных им, с возможностью их фильтрации по статусу. Для этого должны быть отображены следующие поля:
    - Статус жалобы(активна, удовлетворена или отклонена).
    - Дата создания.
    - Текст.
18. Каждому зарегистрированному пользователю должна быть предоставлена возможность просмотра конкретной жалобы. Для этого должны быть отображены следующие поля:
    - Статус жалобы(активна, удовлетворена или отклонена).
    - Дата создания.
    - E mail пользователя, на которого подается жалоба.
    - Телефон пользователя, на которого подается жалоба.
    - Адрес пользователя, на которого подается жалоба.
    - Текстовый комментарий.

- Фотографии.
- Документы.

19. Любому зарегистрированному пользователю должна быть предоставлена возможность оставить отзыв на ту услугу, которую он заказывал. Для этого необходимо заполнить следующие поля:
    - Рейтинг(целое число от 1 до 5, не обязательно для заполнения).
    - Текстовый комментарий(не обязательно для заполнения, максимальное количество символов 500).
    - Фотографии(не больше 5, не обязательно для заполнения).

    При этом приложение должно произвести следующие проверки:
    - Необходимо чтобы был указа или текстовый комментарий, или рейтинг, или оба сразу.
    - Комментарий можно оставить только на услугу, которая была заказана и оказана данному пользователю.

При создании комментария с рейтингом, рейтинг всей услуги необходимо пересчитать(должен быть равен среднему арифметическому рейтингу всех оставленных комментариев)

20. Каждому зарегистрированному пользователю должна быть предоставлена возможность просмотра всех комментариев, созданных им, с возможностью их фильтрации по рейтингу. Для этого должны быть отображены следующие поля:
    - Рейтинг.
    - Текст.
21. Каждому зарегистрированному пользователю должна быть предоставлена возможность просмотра конкретного комментария, созданного им. Для этого должны быть отображены следующие поля:
    - Рейтинг.
    - Текстовый комментарий.
    - Фотографии.
22. Каждому зарегистрированному пользователю должна быть предоставлена возможность обновления конкретного комментария, созданного им. Для этого необходимо заполнить следующие поля:
    - Рейтинг(целое число от 1 до 5, не обязательно для заполнения).
    - Текстовый комментарий(не обязательно для заполнения,

максимальное количество символов 150).

При обновлении комментария с рейтингом, рейтинг всей услуги необходимо пересчитать(должен быть равен среднему арифметическому рейтингу всех оставленных комментариев).

23. Каждому зарегистрированному пользователю должна быть предоставлена возможность удаления конкретного комментария, созданного им. При удалении комментария с рейтингом, рейтинг всей услуги необходимо пересчитать(должен быть равен среднему арифметическому рейтингу всех оставленных комментариев).
24. Каждому пользователю должна быть предоставлена возможность заказа услуги. Для этого необходимо заполнить следующие поля:
    - уникальный идентификатор услуги.
    - дату и время начала выполнения услуги (в формате “год-месяц-день часы:минуты:секунды”).
    - свой телефон (только для незарегистрированных пользователей, но и для них необязательно для заполнения).
    - свой email (только для незарегистрированных пользователей, но и для них необязательно для заполнения, максимальное количество символов 150).
    - свой адрес (не обязательно для заполнения, максимальное количество символов 100).
    - фотографии предполагаемого фронта работ (не больше 5, не обязательно для заполнения).
    - текстовое описание (не обязательно для заполнения, максимальное количество символов 100).

    При этом приложение должно провести следующие проверки:
    - Если услугу заказывает незарегистрированный пользователь, то должен быть указан или телефон, или e mail, или оба.
    - Ни адрес, ни телефон, ни email не должны быть занесены в черный список.
    - Заказ можно сделать только на свободное время (то есть не должно быть других одобренных заказов в промежутке получаса до и после времени начала выполнения услуги).

    В случае успешного создания заказа, пользователю, оказывающему заказанную услугу, должен быть отправлен email с текстом о том, что у него появился новый заказ.
25. Любой пользователь может обновить, сделанный им заказ. Для

этого необходимо заполнить следующие поля:

- дату и время начала выполнения услуги (в формате “год-месяц-день часы:минуты:секунды”).
- свой телефон(только для незарегистрированных пользователей, но и для них необязательно для заполнения).
- свой email(только для незарегистрированных пользователей, но и для них необязательно для заполнения, максимальное количество символов 150).
- свой адрес(не обязательно для заполнения, максимальное количество символов 100).
- фотографии предполагаемого фронта работ(не больше 5, не обязательно для заполнения).
- текстовое описание(не обязательно для заполнения, максимальное количество символов 100).

При этом приложение должно провести следующие проверки:

- Если услугу заказывает незарегистрированный пользователь, то или телефон, или email, или оба должны совпадать с тем, который в заказе.
- Ни адрес, ни телефон, ни email не должны быть занесены в черный список.
- Заказ можно сделать только на свободное время(то есть не должно быть других одобренных заказов в промежутке получаса до и после времени начала выполнения услуги).

26. Любой пользователь может удалить, сделанный им заказ. Если услугу заказывает незарегистрированный пользователь, то или телефон, или e mail, или оба должны совпадать с тем, который в заказе.
27. Зарегистрированному пользователю-мастеру должна быть предоставлена возможность увидеть список заказов на его услуги, с возможностью фильтрации по статусу и сортировки по времени создания и времени начала выполнения заказа. Для этого должны быть отображены следующие поля:
    - дата и время начала выполнения услуги.
    - услуга.
    - дата и время создания заявки на услугу.
    - адрес.
28. Зарегистрированному пользователю-мастеру должна быть

предоставлена возможность увидеть конкретный заказ на его услугу. Для этого должны быть отображены следующие поля:

- дата и время начала выполнения услуги.
- услуга.
- дата и время создания заявки на услугу.
- адрес.
- текстовый комментарий.
- e mail заказчика.
- телефон заказчика.
- фотографии.

29. Зарегистрированному пользователю должна быть предоставлена возможность увидеть список его заказов на услуги, с возможностью фильтрации по статусу и сортировки по времени создания и времени начала выполнения заказа. Для этого должны быть отображены следующие поля:
    - дата и время начала выполнения услуги.
    - услуга.
    - дата и время создания заявки на услугу.
    - адрес.
30. Зарегистрированному пользователю должна быть предоставлена возможность увидеть конкретный свой заказ на услугу. Для этого должны быть отображены следующие поля:
    - дата и время начала выполнения услуги.
    - услуга.
    - дата и время создания заявки на услугу.
    - адрес.
    - текстовый комментарий.
    - e mail заказчика.
    - телефон заказчика.
    - фотографии.
31. Зарегистрированному пользователю-администратору должна быть предоставлена возможность добавлять запись в черный список, для этого необходимо заполнить следующие поля:
    - адрес (не обязательно к заполнению, максимальное количество символов 100).
    - e mail (не обязательно к заполнению, максимальное количество символов 150).

- телефон (не обязательно к заполнению).

При этом приложение должно проверить, чтобы были заполнены или email, или телефон, или они оба. В случае удачного создания записи и если в ней указан e mail, пользователю по этому email, должно быть отправлено письмо, с извещением о том, что он в черном списке.

32. Зарегистрированному пользователю-администратору должна быть предоставлена возможность удалить запись из черного списка. В случае удачного удаления записи и если в ней был указан e mail, пользователю по этому email, должно быть отправлено письмо, с извещением о том, что он больше не в черном списке.
33. Зарегистрированному пользователю-мастеру должна быть предоставлена возможность ответить на полученную заявку, для этого надо заполнить следующие поля:
    - заявка.
    - текст (не обязательно к заполнению, максимальное количество символов 500).
    - статус (может принимать значения “approved” или “rejected”).

В случае удачного создания ответа на заявку и если в заявке был указан email заказчика, по нему должно быть отправлено письмо с извещением о том, что статус его заявки изменился.

34. Зарегистрированному пользователю-администратору должна быть предоставлена возможность ответить на полученную жалобу, для этого надо заполнить следующие поля:
    - жалоба.
    - текст (не обязательно к заполнению, максимальное количество символов 500).
    - статус (может принимать значения “approved” или “rejected”).

В случае удачного создания ответа на жалобу по email-у создателя жалобы должно быть отправлено письмо с извещением о том, что статус его жалобы изменился.

35. Зарегистрированному пользователю-администратору должна быть предоставлена возможность просмотреть все жалобы, на которые он отвечал, для этого должны быть отображены следующие поля:
    - жалоба.
    - текст.
    - статус.
36. Зарегистрированному пользователю должна быть предоставлена

возможность просмотреть все жалобы, которые он создавал, для этого должны быть отображены следующие поля:

- жалоба.
- текст.
- статус.

37. Зарегистрированному пользователю-мастеру должна быть предоставлена возможность просмотреть все ответы на заявки, которые он создавал, для этого должны быть отображены следующие поля:
    - заявка
    - текст
    - статус
38. Зарегистрированному пользователю должна быть предоставлена возможность просмотреть все ответы на заявки, которые он подавал для этого должны быть отображены следующие поля:
    - заявка
    - текст
    - статус

2. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

2.1. Концептуальные сведения

2.1.1.1. Клиент-серверная архитектура

Для веб-приложение, как правило, используется клиент-серверная архитектура. В ее основе лежит два компонента: клиент и сервер.

Клиент это компьютер на стороне пользователя, который отправляет запросы к серверу для выполнения определенных действий или предоставления информации.

Сервер это более мощный компьютер или оборудование, предназначенное для решения определенных задач по выполнению программных кодов, выполнения сервисных функций по запросу клиентов, предоставления пользователям доступа к определенным ресурсам, хранения информации и баз данных.

Эта архитектура заключается в том, что клиент формирует необходимый запрос на получение данных или изменения их и отправляет его серверу. На сервере этот запрос обрабатывается и ответ возвращается обратно к клиенту.

Функции, которые выполняет сервер:

- обработки клиентских запросов;
- оправка результатов клиенту
- хранение данных, а также их защита и определение прав доступа к ним;

Функции, которые выполняет клиент:

- создание запроса к серверу и его отправка;
- предоставление интерфейса пользователя;
- получение результата запроса от сервера

Клиент и сервер взаимодействуют между собой с помощью различных сетевых протоколов. Для взаимодействия, как правило, используется протокол HTTPS, который в свою очередь является расширением протокола HTTP с поддержкой шифрования.

2.1.2. REST API

API (Application Programming Interface) — это интерфейс программирования, интерфейс создания приложений. RESTful API - это архитектурный стиль для интерфейса прикладных программ (API), который использует HTTP-запросы для доступа и использования данных. REST, используемый браузерами, можно рассматривать как язык

Интернета. API RESTful используются такими сайтами, как Amazon, Google, LinkedIn и Twitter.

RESTful API использует команды для получения ресурсов. Состояние ресурса в любой момент времени называется представлением ресурса. RESTful API использует существующие HTTP-методологии, определенные протоколом RFC 2616, например:

- GET для получения ресурса;
- PUT, PATCH для изменения состояния или обновления ресурса, который может быть объектом, файлом или блоком;
- POST для создания этого ресурса;
- DELETE, чтобы удалить его.

2.1.3. MVT

Фреймворк Django реализует архитектурный паттерн Model-View-Template или сокращенно MVT, который является модификацией распространенного паттерна MVC (Model-View-Controller). На рисунке 1.1 представлена схематичная архитектура данного паттерна.

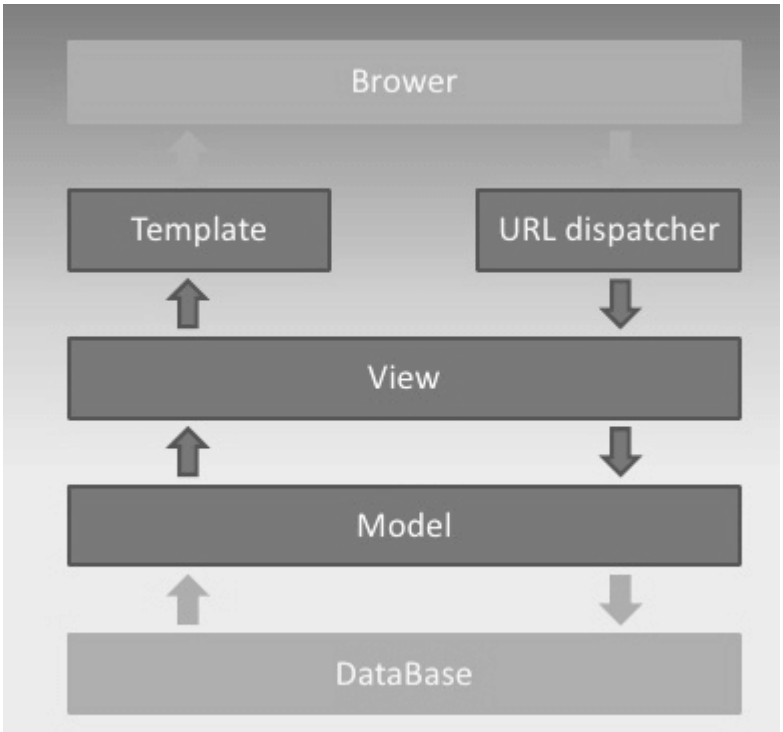

Рисунок 1.1 –Структура паттерна MVT.

Однако для написания только back end части приложения шаблоны

не нужны. Кроме того, для написания клиентской части веб-приложения часто используют специализированные языки программирования и фреймворки. С бэк-эндом, написанном на python, такие фреймворки взаимодействуют по api, поэтому для написания используется фреймворк, который называется Django REST Framework (или, сокращенно, DRF). Основная архитектура DRF состоит из 3-х слоев: сериализатора, вида и маршрутизатора.

Сериализатор: преобразует информацию, хранящуюся в базе данных и определенную с помощью моделей Django, в формат, который легко и эффективно передается через API.

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

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

Сериализаторы(Serializers)

Модели Django интуитивно представляют данные, хранящиеся в базе, но API должен передавать информацию в менее сложной структуре. Хотя данные будут представлены как экземпляры классов Model, их необходимо перевести в формат JSON для передачи через API.

Сериализатор DRF производит это преобразование. Когда пользователь передает информацию (например, создание нового экземпляра) через API, сериализатор берет данные, проверяет их и преобразует в нечто, что Django может сложить в экземпляр модели. Аналогичным образом, когда пользователь обращается к информации через API, соответствующие экземпляры передаются в сериализатор, который преобразовывает их в формат, который может быть легко передан пользователю как JSON.

Наиболее распространенной формой, которую принимает сериализатор DRF, является тот, который привязан непосредственно к модели Django. Сериализаторы — это невероятно гибкий и мощный компонент DRF. Хотя подключение сериализатора к модели является наиболее распространенным, сериализаторы могут использоваться для создания любой структуры данных Python через API в соответствии с определенными параметрами.

Виды (Views)

Сериализатор анализирует информацию в обоих направлениях (чтение и запись), тогда как ViewSet - это тот код, в котором определены

доступные операции. Наиболее распространенным ViewSet является ModelViewSet, который имеет следующие встроенные операции:

- Создание экземпляра: create ()
- Получение / чтение экземпляра: retrieve ()
- Обновление экземпляра (все или только выбранные поля): update () или partial_update ()
- Уничтожение / Удаление экземпляра: destroy ()
- Список экземпляров: list ()

Каждая из этих функций может быть переопределена, если требуется другое поведение, но стандартная функциональность работает с минимальным кодом. Если необходимы дополнительные настройки, можно использовать общие представления, вместо ModelViewSet или даже отдельные пользовательские представления.

Маршрутизаторы (Routs)

И наконец, маршрутизаторы: они предоставляют верхний уровень API. Чтобы избежать создания бесконечных URL-адресов вида: «списки», «детали» и «изменить», маршрутизаторы DRF объединяют все URL-адреса, необходимые для данного вида в одну строку для каждого View.

2.1.4. ORM

ORM(Object-relational-mapping) - это идея возможности писать запросы к базе данных, используя объектно-ориентированную парадигму используемого языка программирования. Короче говоря, это взаимодействие с базой данных, используя предпочитаемый язык вместо SQL. У такого подхода есть свои как плюсы, так и минусы. Плюсами являются:

- Возможность работать с базой данных практически не зная языка SQQ
- Абстрагирование от непосредственной СУБД, что значительно облегчает переход с одной из них на другую в случае необходимости
- В зависимости от используемой ORM, получение расширенных функций (например миграции в Django)
- Оптимизации запросов

У этого подхода также имеются и отрицательные стороны:

- Первоначальная настройка ORM часто требует достаточно много времени

- Необходимость знать особенности работы используемой ORM

Несмотря на имеющиеся недостатки, достоинства данного подхода все же являются достаточно значительными, что способствует широкому использованию данной технологии.

2.2. Язык программирования Python

Одним из преимуществ языка Python является кроссплатформенность. Python – это интерпретируемый язык, его интерпретаторы существуют для многих платформ. Поэтому с запуском его на любой ОС не должно возникнуть проблем. С Python доступно огромное количество сервисов, сред разработки, и фреймворков. Легко можно найти подходящий продукт для работы.Возможность подключить библиотеки, написанные на C. Это позволяет повысить эффективность, улучшить быстродействие. Python отличается строгим требованием к написанию кода (требует отступы), что является преимуществом. Изначально язык способствует писать код организованно и красиво. Для большинства задач: для веб-разработки, для скриптов, прототипирования, машинного обучения и работы с большими данными, — один из лучших языков. Все эти преимущества способствуют тому, что в качестве основного языка для написания back-end для сервиса будет выбран именно Python.

2.3. Фреймворк Django

Django является очень популярным серверным веб-фреймворком, который написан на Python. Фреймворк — это набор компонентов, которые помогают разрабатывать веб-сайты быстро и просто. Один из основных принципов фреймворка — DRY (don’t repeat yourself). Также, в отличие от многих других фреймворков, обработчики URL в Django конфигурируются явно (при помощи регулярных выражений), а не автоматически задаются из структуры контроллеров.

Django проектировался для работы под управлением Apache и с использованием PostgreSQL в качестве базы данных. В настоящее время, Django также может работать с другими СУБД: MySQL, SQLite, Microsoft SQL Server, DB2, Firebird, SQL Anywhere и Oracle. Для работы базой данных Django использует собственный ORM, в котором модель данных описывается классами Python, и по ней генерируется схема базы данных. Веб-фреймворк Django используется в таких крупных и известных сайтах,

как Instagram, Disqus, Mozilla, The Washington Times, Pinterest, lamoda и др.

Фреймворк Django имеет множество достоинств, некоторые из них:

1. Django содержит большое количество функционала, который существенно упрощает веб-разработку. Примерами являются ORM, аутентификация, миграции базы данных,возможность подключения панели администратора и тд.
2. Django как фреймворк сам уже задает структуру проекта, что позволяет разработчикам гораздо проще находить нужные им файлы с кодом, а также упрощает поддержку уже существующих проектов.
3. Django достаточно безопасен из коробки, так как содержит механизмы предотвращения распространенных атак, таких как подделки межсайтовых запросов или SQL-инъекций (XSS).
4. Django REST Framework, который так же часто называю«DRF», является широко библиотекой для построения API. Он позволяет достаточно быстро и просто создавать как простые, так и довольно сложные API. Значительным плюсом является также то, что он поставляется со встроенной возможность тестирования API и базовыми классами для CRUD операций.

2.4. PostgreSQL

PostgreSQL - это мощная система объектно-реляционной базы данных с открытым исходным кодом, которая использует и расширяет язык SQL в сочетании с множеством функций, обеспечивающих безопасное хранение и масштабирование данных. Истоки PostgreSQL восходят к 1986 году как часть проекта POSTGRES в Калифорнийском университете в Беркли и более 30 лет активно развиваются на базовой платформе.

PostgreSQL заработал прочную репутацию благодаря своей проверенной архитектуре, надежности, целостности данных, надежному набору функций, расширяемости и приверженности сообщества разработчиков ПО с открытым исходным кодом к последовательной разработке эффективных и инновационных решений. PostgreSQL работает во всех основных операционных системах, поддерживает ACID с 2001 года и имеет мощные надстройки. Неудивительно, что PostgreSQL стала реляционной базой данных с открытым исходным кодом, которую выбирают многие разработчики и организации.

Основными причинами выбрать для написания дипломного проекта PostgreSQL были широкая поддержка различных типов данных, которыми не обладают другие реляционные СУБД, а так же хорошая поддержка Django именно PostgreSQL, тк фреймворк изначально писался для использования именно этой СУБД и оптимизирует многие запросы, а также существует отдельная библиотека для работы с PostgreSQL, которая позволяет полностью использовать функционал, специфичный для это СУБД.