МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Казахский национальный исследовательский технический университет имени К.И.Сатпаева

Институт автоматики и информационных технологий

Кафедра "Программная инженерия"

Дукенбаев Дален Ерболулы

Разработка портала фрилансеров

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к дипломному проекту

По специальность 5В070400 – Вычислительная техника и программное обеспечение

Алматы 2022

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Казахский национальный исследовательский технический университет имени К.И.Сатпаева

Институт автоматики и информационных технологий

Кафедра "Программная инженерия"

ДОПУЩЕН К ЗАЩИТЕ
Заведующий кафедрой ПИ
канд. техн. наук, доцент,
профессор
_____Молдагулова А.Н.
"__17__" ____05_____ 2022 г.

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к дипломному проекту

На тему: Разработка портала фрилансеров

по специальность 5В070400 – Вычислительная техника и программное обеспечение

Выполнил
Рецензент
Директор Института ИТ «АУЭС»
_____А.А. Досжанова
"_____"_____ 2022 г.

Дукенбаев Дален Ерболұлы
Научный руководитель
Ассоциированый профессор
_____Н. К. Мукажанов
“____” _____2022 г.

Алматы 2022

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Казахский национальный исследовательский технический университет имени К.И.Сатпаева

Институт автоматики и информационных технологий

Кафедра "Программная инженерия"

5В070400 – Вычислительная техника и программное обеспечение

УТВЕРЖДАЮ
Заведующий кафедрой ПИ
канд. техн. наук, доцент,
профессор
_____Молдагулова А.Н.
"17" 05 2022 г.

ЗАДАНИЕ
на выполнение дипломного проекта

Обучающемуся *Дукенбаеву Далену Ерболулы*
Тема: *Разработка портала фрилансеров*
Утверждена приказом проректора по академической работе: № 489-П/О
от "24" 12 2021 г.
Срок сдачи законченного проекта: "24" 05 2022 г.
Исходные данные к дипломному проекту: сбор теоретического материала, данные анализа по данной теме
Перечень подлежащих разработке в дипломном проекте вопросов:
А) Анализ предметной области, исследования рынка, актуальность проекта, составление и постановка задач;
Б) Выбор инструментов проектирования и разработки системы;
В) Проектирование системы;
Г) Реализация системы, создания пользовательского интерфейса и разработка системы;
Д) Написание тестовых наборов данных и тестирования.
Перечень графического материала (с точным указанием обязательных чертежей): *представлены 29 слайда презентации.*
Рекомендуемая основная литература: *из 20 наименований.*

ГРАФИК

подготовки дипломного проекта

Наименование разделов, перечень разрабатываемых вопросов	Сроки представления научному руководителю и консультантам	Примечание
1. Анализ предметной области, разработка технического задания на проектирование архитектуры системы	20.01.2022	Выполнил
2. Выбор инструмента для ведения разработки системы.	30.01.2022	Выполнил
3. Написание технического задания проекта	10.02.2022	Выполнил
4.Создание прототипа портала с помощью редактора «Figma»	18.02.2022	Выполнил
5. Верстка проекта с сервиса «Figma» на языке разметки страницы «HTML» и языка описания внешнего вида «CSS». (Front-end разработка)	28.02.2022	Выполнил
6. Осуществление разработки функционирования внутренней части портала на языке программирования Python, фреймворк Django (Back-end разработка)	24.03.2022	Выполнил
7.Тестирование проекта	28.03.2022	Выполнил

Подписи

консультантов и нормоконтролера на законченный дипломный проект с указанием относящихся к ним разделов проекта

Наименования разделов	Консультанты,И.О.Ф. (уч. степень, звание)	Дата подписания	Подпись
Нормоконтроллер	Жекамбаева М.Н. Доктор Ph.D, ассоциированный профессор	16.05.22г	
Программное обеспечение	Марғұлан. К. Магистр техн.наук, лектор	16.05.22г	

Научный руководитель _____ Н. К. Мукажанов

Задание принял к исполнению обучающийся _____ Д.Е. Дукенбаев

Дата «17» 11 2021 г.

АННОТАЦИЯ

Данное Web-приложение разработано для пользователей по поиску удаленной работы в Республике Казахстан, целью которой является предоставление взаимовыгодных отношений и общению между фрилансером и заказчиком. Дипломный проект состоит из трех главных разделов, а также из заключения и списка используемой литературы.

Первый раздел включает в себе общее описание, а также посвящен терминам и сокращениям, целям и задачам, которые предстояло реализовывать.

Во втором разделе полностью описывается рабочая среда, архитектура Фреймворка и обосновывается выбранный инструмент для ведения разработки и почему выбран тот или иной подход разработки.

Третья часть проекта посвящена структуре и архитектуре проекта.

В заключении подводится итог всей работы.

АҢДАТПА

Бұл Web-қосымша Қазақстан Республикасында қашықтықтан жұмыс іздеу бойынша пайдаланушылар үшін әзірленген, оның мақсаты фрилансер мен Тапсырыс беруші арасындағы өзара тиімді қатынастар мен қарым-қатынасты ұсыну болып табылады. Дипломдық жоба үш негізгі бөлімнен, сондай-ақ қорытынды мен пайдаланылған әдебиеттер тізімінен тұрады.

Бірінші бөлім жалпы сипаттаманы қамтиды, сонымен қатар терминдер мен қысқартуларға, іске асырылуы тиіс мақсаттар мен міндеттерге арналған.

Екінші бөлімде жұмыс ортасы, жақтау архитектурасы толық сипатталған және дамуды жүргізу үшін таңдалған құрал негізделген және дамудың осы немесе басқа тәсілі неге таңдалған.

Жобаның үшінші бөлімі жобаның құрылымы мен сәулетіне арналған.

Қорытындылай келе, барлық жұмыс қорытындыланады.

ANNATATION

This Web application is designed for users to search for remote work in the Republic of Kazakhstan, the purpose of which is to provide a mutually beneficial relationship and communication between the freelancer and the customer. The diploma project consists of three main sections, as well as the conclusion and a list of references used.

The introduction discusses the reason for automating the Faculty Competition.

The first section includes a general description and is devoted to the terms and abbreviations, goals and objectives to be implemented.

The second section fully describes the working environment, the architecture of the Framework and justifies the chosen development tool and why a particular development approach was chosen.

The third part of the project is devoted to the structure and architecture of the project.

The conclusion summarizes all the work.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Актуальность темы моего дипломного проекта заключается в том, что в мире, где технологический мир не стоит на месте, где запросов и заказов людей, владеющими каким либо бизнесом, к примеру создания того же простого логотипа, появляются с каждым днем все больше и больше, а вопросы, где найти специализированного человека, разбирающегося в их задаче, все сложнее и сложнее, особенно если, часто вопрос стоит в доверии между людьми. Хоть в данный момент существуют множество сервисов, которые предоставляю возможность найти хорошего специалиста и могут привести их к выгодным взаимоотношениям и партнерским соглашениям, но в целом данная тема в нашей стране почти никак не развивается, тем более если учесть тот факт, что огромное количество подобных иностранных сайтов блокируют.

Объект исследования сервисы самостоятельного контроля своего дня и решения интересных задач, заказчиков, со стороны разработчика.

Предмет исследования данного дипломного проекта стала «Фриланс-биржа», сервисы, являющиеся уполномоченными посредниками между заказчиками, то бишь обычными пользователями и фрилансерами, являющимися специалистами в какой-либо научной, технической и художественной области.

Цель дипломной работы разработка web-приложения, портала для фрилансеров, задачи, которые необходимо достичь в ходе разработки софта:

- изучение и исследование рынка;
- определение требований к системе;
- разработка архитектуры системы;
- разработка приложения;
- тестирования системы.

1. Общее представление о проекте

1.1 Цель разработки системы

Основная цель разработки системы является создание своего портала фрилансера. Так как это будет будущим сайтом, то нужно дать ему простое и запоминающее наименование, для дальнейшей разработки. Для своего портала фрилансеров, я выбрал имя «TaskStart», которое будет является доменным именем после развертывания сайта на хостинг.

1.2 Анализ рынка

Пандемия 2019 года охватила весь мир и изменила его так сильно, что поменяла всеми нами любимую и стабильную жизнь, на огромное количество ограничений и запретов, так или иначе, полностью перевернув мировой рынок труда, огромное число людей остались без работы, но с сильным рвением жить и адаптируясь к новому порядку существования, где интернет стал важной частью жизни и переходом на удаленную деятельность.

Для того же большинства людей, выходом стал фриланс. Казахстан в свою очередь тоже не стал стоять в стороне.

Фриланс-биржа — это сервисы, которые предоставляют возможность удаленно работать, получать заказы, выполнять сложные и интересные задачи в разных сферах, размещать свои услуги на онлайн-площадках, тем самым ища потенциальных заказчиков, в поисках исполнителя для своих проектов.

Хоть фриланс и идет по всей планете, привлекая к себе все больше и больше людей, у нас в Казахстане, к сожалению, после ослабления всех ограничений и запретов, поиск удаленной работы или фриланса упал, что можно разглядеть на статистике проведенную по запросам слов «фриланс» и «удаленная работа», к примеру с 2020 года по сегодняшний день.

Рисунок-1.1 – Статистика запросов поиска по слову «фриланс»

Связано это с тем, что на таких площадках достаточно тяжело зарабатывать много, так же, большую роль играет построение доверительных отношений между клиентом и фрилансером, хоть и портал играет роль посредника, но не всегда может предотвратить обман. Так как в Казахстане придерживаются мнения, что, работая на постоянной работе с проверенными людьми, можно получить намного больше опыта, несмотря на то что на фрилансе, наработать опыт и собрать отличное портфолио намного легче.

1.2.1 Анализ по предметной области

Порталы являются разновидностью веб-сайтов, функционал которого, гораздо больше и навигация удобнее. Данный вид сайта ориентированы на определенную тематику и объединяют вокруг себя пользователей с одинаковыми интересами.

Портал для фрилансеров – сайт, предоставляющий возможность удаленной работы разработчикам, дизайнерам и другим специалистам, так же является посредником между фрилансером и клиентом.

Пройдя регистрацию пользователь уже может, воспользовавшись услугой сайта по размещению контента, выставить задачу на сайт и стать фрилансером. Выставленные задачи на сайте и вся информация хранится в базе данных. Потенциальный клиент, зайдя на полную информацию заказа, может, как узнать о предоставленных услугах фрилансером, так и о самом разработчике. После выбора задачи, удовлетворяющие все условия пользователя, он может написать фрилансеру находясь на сайте и договорится о цене и условиях заказа.

Сайт создается с применением систем управления контентом. Каждый крупный портал работает, уже со специально разработанными системами управления и слежением за происходящим на сайте. Обычно средние или

малые проекты используют свободное ПО, к примеру как OpenCart, Drupal, WordPress и т. д.

Система управления сайтом можно установить самостоятельно на хостинг-площадку, могут быть частной разработкой или ПО с временными подписками.

Существуют два вида порталов по поиску удаленной работы:

1) Сайты с вакансиями по поиску удаленных работ, где работодатели выставляют посты с вакансиями, в которых ищут сотрудников на удаленную работ.

2) Сайты биржи фриланса, где разработчик сам работает на себя и выставляет на портал задачи, которые он может решить, контролируя свое свободное время и выполнять работы заказчиков.

1.2.2 Анализ по существующим разработкам

Существуют огромное количество проектов по бирже фриланса, которые на столько закрепились на данной позиции, что при одном только упоминании о фрилансе, их сайты появляются в поисковых результатах.

На данный момент не сложно найти наш список бирж фриланса, к примеру такие как:

- allfreelance.kz;
- ozat.kz;
- megamaster.kz/freelance;
- freelancehunt.kz;
- uwork.kz.

Несмотря на то, что мы имеем свои сайты для фрилансеров, хороший специалист не должен ограничивается в поиске работы только по казахстанским биржам, это будет не эффективно и неправильно.

Взяв во внимание и проанализировав сайты из списка, можно выделить следующий основной функционал, который связывает их всех и недостатки, которые так же их объединяют.

Общий основной и базовый функционал сайтов по фрилансу, заключается в наличии:

- регистрации и авторизации;
- просмотр локальной информации по теме портала;
- простая и удобная навигации по сайту;
- поисковая система;
- создание задач и заказов для фрилансеров и клиентов;
- просмотр задач и заказов;
- возможность писать отзывы к заказам;
- личные сообщения между пользователями сайта.

Так же можно выделить незначительные и грубые недостатки на данных сайтах, а именно:

- сложная навигация по сайту;
- сложный и в некоторых местах кривой интерфейс;
- слишком много условий для начала работы, такие как: подписка, ввод лишних данных, много появляющейся рекламы и информации.

2. Определение требований к системе

Для данного проекта стоит уделить высокое внимание требований к ИС, до начала разработки, написав требования, как часть технического задания.

Мы должны определить следующие требования к системе:

- функциональные требования;
- требования к интерфейсу;
- требования к производительности;
- требования к безопасности.

2.1 Функциональные требования

Функциональные требования определяют методы и функции системы, которую мы создаем для пользователей, предоставляя им функциональные возможности приложения, для дальнейшего использования.

ИС должна обеспечивать следующие функциональные возможности:

- регистрация новых пользователей;
- авторизация существующих пользователей;
- сброс пароля и смена его на новый;
- изменение логина и почты пользователя;
- редактирования профиля пользователя;
- создания заказов для фрилансеров;
- редактирование заказа, написание новой или дополнительной информации к существующей;
- удаление существующего заказа;
- создание задач для потенциальных заказчиков;
- редактирование задач, написание новой или дополнительной информации к существующей;
- удаление существующих задач;
- писать отзывы к существующим заказам и задача;
- писать отзывы на странице профиля другого пользователя;
- создания формы запроса обратной связи и помощи.

2.2 Требования к интерфейсу

Под интерфейсом подразумевается тип экранного представление, при котором пользователь может осуществлять выбор предоставленных ему функций, запускать какие-либо представления и просматривать контент.

Проектирование интерфейса для портала должен соответствовать общему представлениям и согласованным нормам, а также по всем правилам создания пользовательских интерфейсов:

- специально подобранная под тематику портала цветовая гамма;
- простой и ясный порядок выполнения шагов и действий на сайте;
- удобная и понятная навигация по сайту;
- эргономичное расположение полей ввода и элементов управления.

ИС должна иметь следующий интерфейс:

- форма регистрации и авторизации пользователя;
- форма редактирования профиля пользователя;
- отображение существующих задач и заказов;
- отображение отзывов на существующих заказах, комментариях и профилях зарегистрированных пользователей;
- форма создания заказа – возможность записать всю нужную информацию к заказу, дополнительную и загрузка картинки для внешнего вида заказа;
- форма редактирования заказа;
- форма создания задачи – возможность записать всю нужную информацию к заказу, дополнительную и загрузка картинки для внешнего вида заказа;
- форма редактирования задачи;
- строка поиска по порталу – должна быть видна и позволять задавать критерии для поиска;
- простые и понятные шапка портала и подвал, для навигации по страницам.

2.3 Требования к производительности

Как и любое web-приложение, сайт должен иметь требование к эффективности и производительности. В разных системах имеются свои процессы оптимизации, которые состоят из слоев системы, каждый из которых имеет свой комплект ограничений. Здесь описаны основные характеристики оптимизации производительности:

- задержки и пропускная способность;
- ограничения TCP протокола;
- недостатки протокола HTTP;
- ограничения браузера;
- кэширование web-приложения.

Далее требования, подходящего для портала для фрилансеров:

- приложение должно сохранять стабильную работу при одновременном доступе нескольких пользователей;

- приложение не должно занимать большой объем памяти, если на сайте малое количество пользователей;
- база данных сайта должна быстро обслуживать запросы, даже при наличии нескольких приложений;
- кэширование шаблонов web-страниц;
- приложение чата для пользователей должно быстро реагировать на запросы отправки сообщений пользователями другим пользователям;
- оптимизация на серверной части кода.

2.4 Требование к безопасности

Безопасность – служит покоем для многих пользователей интернета и порой вызывает страшные последствия в случае проблем. Безопасность сайта — это защита от перебоев производительности, DDoS-атак, потерь данных из базы данных и потерь личной информации пользователя. Нужно серьезно относится к данному вопросу, обеспечить хорошую защиту и выбрать надежный хостинг, так же написать требования к безопасности, которые могли бы сократить проблемы вызванными недоброжелательными личностями.

Требования к безопасности к порталу можно выделить следующие:
- поставить отличную защиту от DDoS-атак;
- выбрать надежный хостинг для портала;
- иметь хэширование паролей пользователей;
- обеспечить надежным паролем супер-пользователя;
- создать мониторинг безопасности и следить за активностью сайта на админ-панели;
- установить бэкап сайта.

3. Проектирование

3.1 Архитектура системы

Во время создания системы и принятия проектного решения, для проекта важно описать архитектуру системы, модель, которая собирает в себе концепцию многих структур, где та в свою очередь описывает взаимодействие между структурами и действия их в системе. Так как мы имеем web-приложение то и модель архитектуры должна быть соответсвенная и должна определять, насколько быстрым, безопасным, чутким и прочным будет портал.

Архитектура описывает следующие структуры моего web-приложения:

- клиентский PC;
- DNC;
- CDN;
- балансировщик нагрузки;
- сервер;
- клиентское приложение;
- сервер генерации;
- панель Администрации.

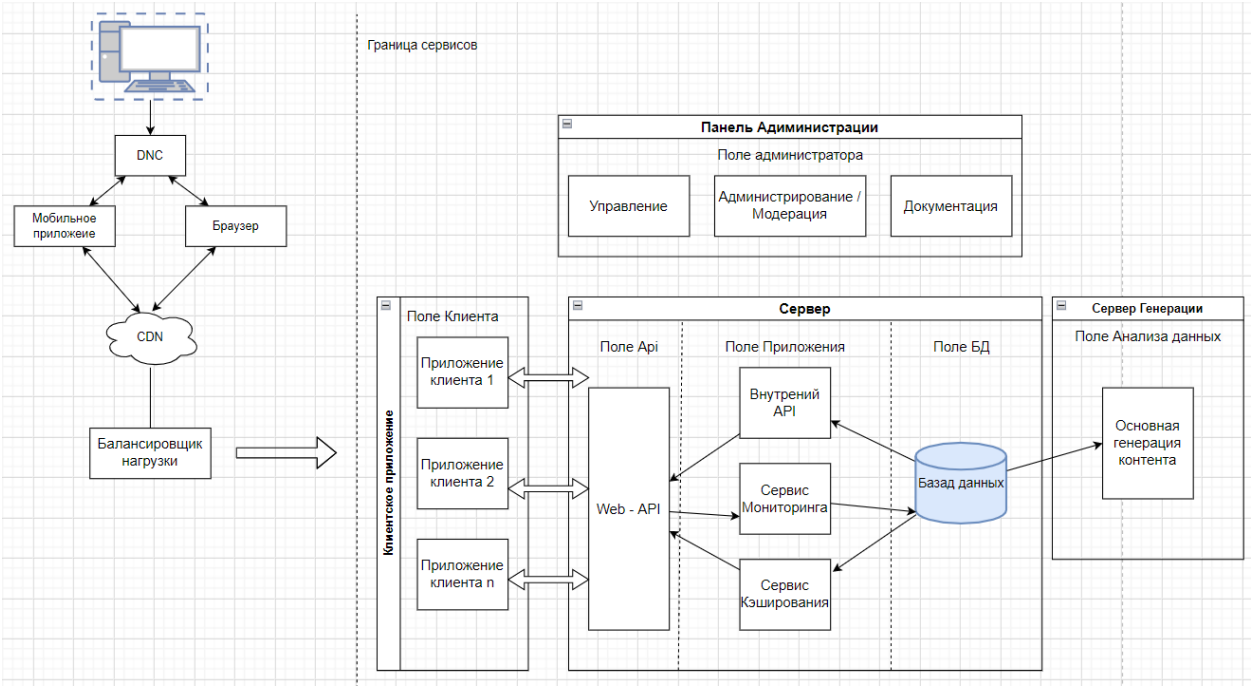

Рисунок-3.1 – Архитектура системы web-приложения «Портал для фрилансеров»

3.2 Проектирование модели базы данных

Логическая модель базы данных моего портала будет описывать схему базы данных. Сущности в ней представляют типы объектов, которые хранятся в БД.

В начале проектирования ИС, нужно выбрать количество основных сущностей, модели базы данных портала:

- пользователи (User) – сущность, предоставляющая полную информацию о каждом зарегистрированном пользователе;
- задачи (Task) – сущность, представляет информацию о задачах, которые создают фрилансеры;
- заказы (Post) – сущность, представляет информацию о заказах, созданные работодателями или заказчиками;
- комментарии пользователей (Comments) – являются сущностью комментариев пользователей, хранит в себе отзыв о задачах, заказах или о другом пользователе;
- профиль пользователя (Profile) – сущность, связанная с пользователем, представляет подробную информацию о пользователе;
- рубрика (Main category) – сущность, представляет основные рубрики, которые принадлежат задачам и заказам;
- подрубрика (Category) – сущность, так же связанная с задачами и заказами, которые пользователь присваивает им при создании;
- мессенджер (Message) – сущность, которая хранит в себе данные о переписке между пользователями, хранит их личные сообщения;

Далее у каждой сущности есть атрибуты, которые следуют определить в проекте. Для пользователя определенны следующие атрибуты:

- ключевой атрибут для каждой сущности (id или user_id) - идентификационный номер пользователя;
- логин (login) – для регистрации и будущей авторизации пользователей и входа в систему;
- пароль (Password) – пароль;
- почта (email) – почта пользователя, записывается при регистрации и играет важную роль при сбросе пароля;
- супер-пользователь (root) – права администратора, обычно либо true или false;
- идентификационный номер для профиля пользователя (profile_id) – представляет дополнительную информацию о пользователе;

Далее для задачи, которые создает фрилансер, существуют следующие атрибуты:

- идентификационный номер задачи (task_id) – ключевой атрибут сущности задачи, который создает фрилансер;
- наименование (title) – название задачи;

- наименование адреса задачи (slug) – у каждой задачи существует свое второе наименование для записи его позже в адресную строку созданной задачи;

- описание (content) – подробная информация о задаче, в которой можно будет представить требования к выполнению условий;

- цена (price) – текущая цена за услуги, описанные в задаче;

- превью заказа (photo);

- рубрика задачи (main cat id);

- категория задачи (cat id).

Сущность заказов созданные пользователем будут иметь те же атрибуты. Так же сущность рубрики и категорий имеют всего по 4 атрибута, это id, title, id задачи и id заказа, что так же и про комментарии пользователей, а именно их отзывы имеют 4 атрибута user id, task id, post id и сам отзыв (content).

Сущность профиля пользователя имеет следующие атрибуты:

- идентификационный номер профиля (id);

- идентификационный номер пользователя (user id);

- имя пользователя (name);

- фамилия пользователя (surname);

- ссылки на социальные сети;

- номер телефона (phone number);

- информация о пользователе (about);

- рейтинг пользователя (rating);

- знание языков (language);

- профессия или опыт работы (experience);

- фотография пользователя (avatar).

сущность мессенджера имеет следующие атрибуты:

- идентификационный номер чата с пользователем (id);

- идентификационный номер пользователя – атрибут нужен для распределения сообщений между первым и вторым пользователем в их диалоге;

- идентификационный номер первого пользователя (first user id);

- идентификационный номер второго пользователя (second user id);

- текст сообщения (body);

- дата сообщения (date);

- уведомление о новом сообщении (is read).

Далее идут связи между сущностями, где те играют важную роль между таблицами в базе данных.

Пользователь может создать несколько задач и несколько заказов, поэтому между этими сущностями связь один ко многим (1:M), так же сущность пользователя может иметь лишь один профиль и профиль может быть только у одного пользователя, отсюда следует связь один к одному (1:1).

Комментарии (Comment) к заказам, задачам и к профилю пользователя могут написать множество пользователей, поэтому связь будет многие ко многим (M:M).

Рубрики и категории являются отдельными сущностями, так как в ходе обновления контента, можно будет добавить новую рубрику или категорию и отобразить уже с базы данных на сайте. Рубрики и категории могут быть у многих задач и заказов, но только у одной задачи и заказа может быть одна рубрика и одна категория, поэтому связь будет многие к одному или один ко многим (1:M).

Сообщения или же мессенджер привязан к всем пользователям, так же как и у всех клиентов есть возможность пользоваться мессенджером и взаимодействовать с друг другом, поэтому связь многие к многим (M:M).

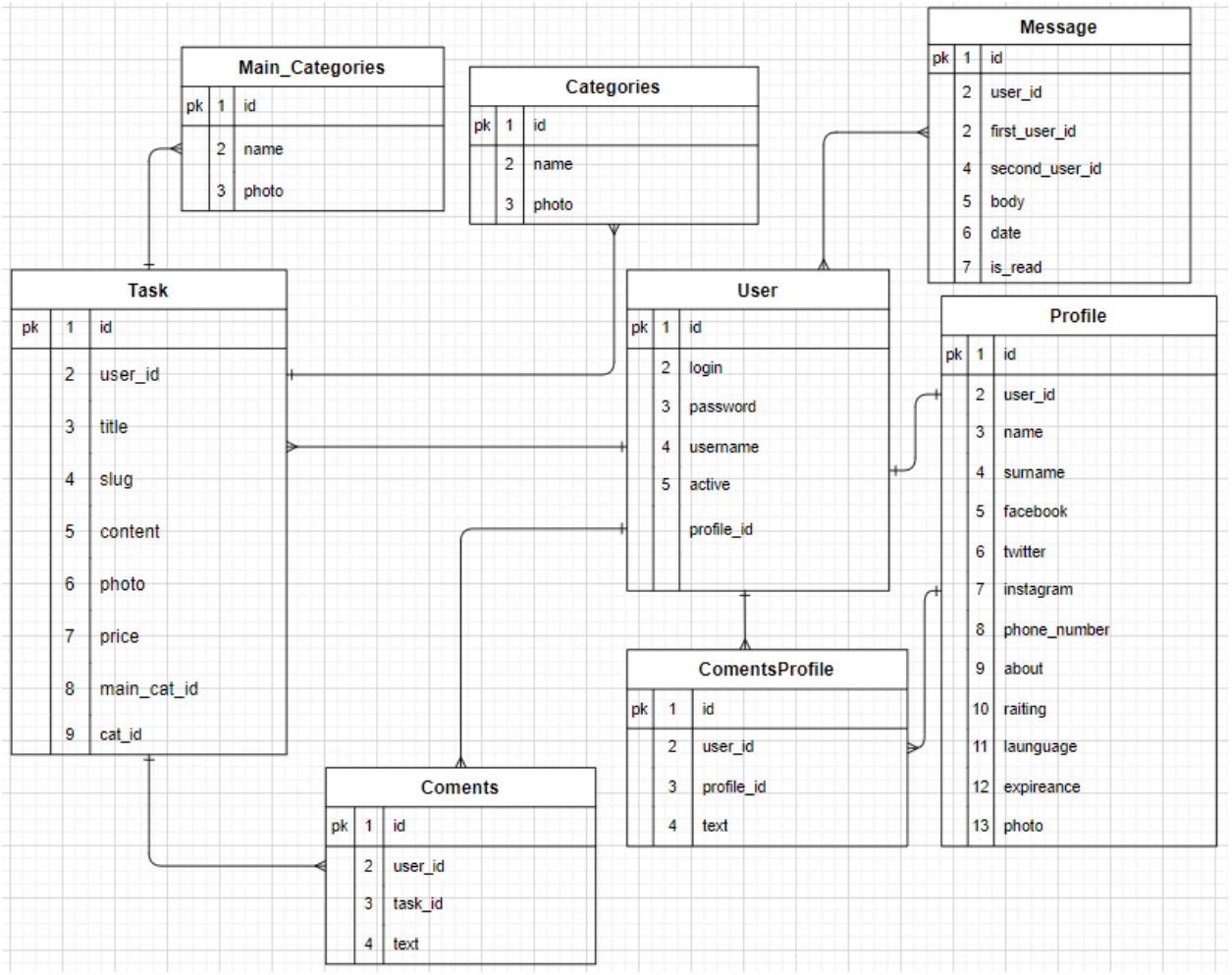

Рисунок-3.2 – ER диаграмма базы данных и связи между сущностями

3.3 UML модель

Каждый раз, начиная задумывается о новом проекте и начиная писать код, мы могли потерять часть той целой картины, нашей идеи, чтобы этого избежать существуют - UML диаграммы.

UML (Unified Modeling Language) – представляет собой диаграммы для моделирования и проектирования систем и играют очень важную роль в разработке проектов ООП.

3.3.1 Use-case диаграмма

Первым с чего следует начать это то, кто будет пользоваться приложением и что они могут сделать, а именно им нужно поставить варианты использования. Диаграмма вариантов использования (Use-case диаграмма) – представляет возможности функционала.

Портал для фрилансеров имеет следующие группы пользователей:

- клиент;
- фрилансер.

В основном Клиент и Фрилансер это тот же Пользователей и имеют они схожие варианты использования сайта.

Пользователь может:

- просматривать любую статическую информацию на сайте;
- завести свой аккаунт, то есть зарегистрироваться на портале;
- авторизироваться на сайте;
- создать свой личный кабинет и редактировать профиль;
- после регистрации, будет возможность просматривать динамический контент на сайте;
- использование чата, для общения с другими посетителями сайта;
- при появлении вопроса или бага на сайте, есть возможность обратной связи;

Если пользователь будет в роли клиента, то его действия:

- создание своего заказа;
- просмотр задач, созданных фрилансерами;
- возможность написания отзывов к задачам и написания отзывов в чужих профилях.

Если пользователь будет в роли фрилансера, то его действия:

- создание своего задачи;
- просмотр заказов, созданных клиентами.

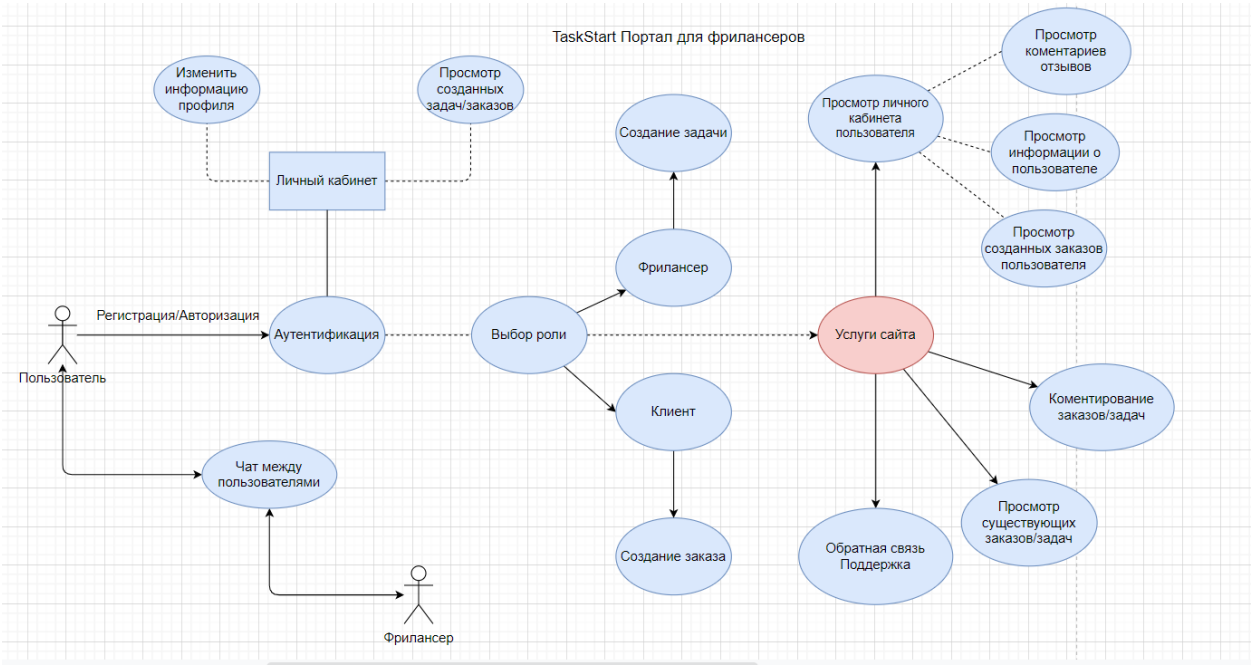

Рисунок-3.3 – Use-case диаграмма «UML модель»

3.3.1 Диаграмма классов

Создание классов при разработке своего приложения, это обязательная часть любого проекта. Диаграмма классов позволяет определить связи между классами и подробно расписать методы внутри классов и наглядно показать атрибуты внутри них.

Классы представления данных, показаны на диаграмме ниже:

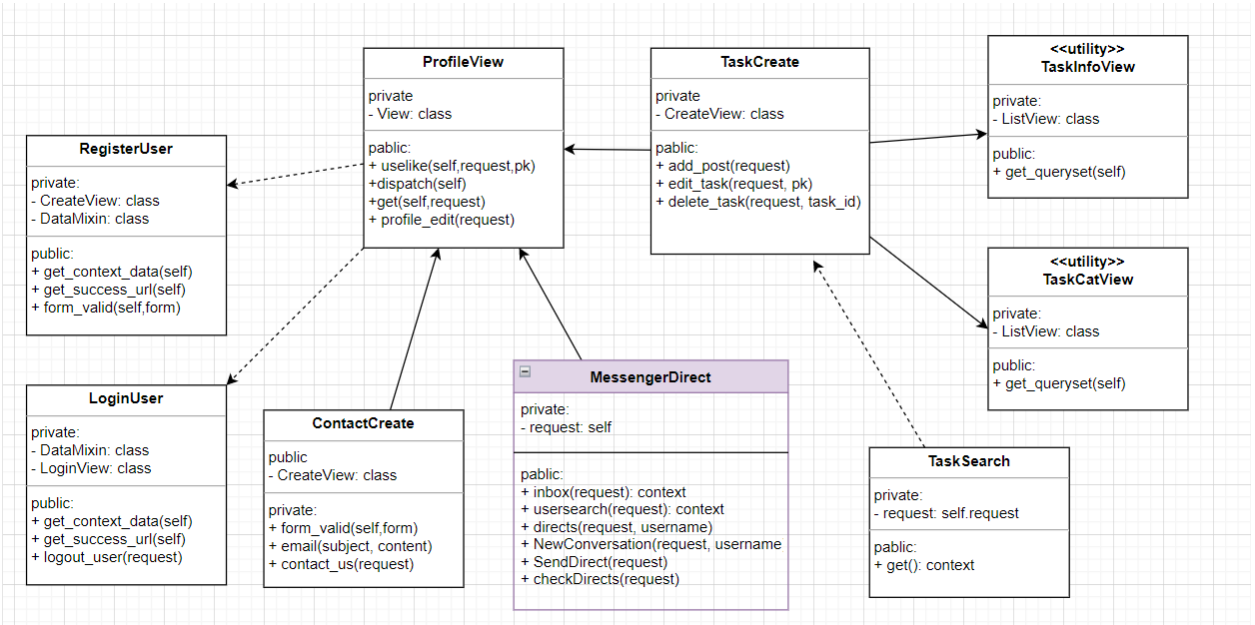

Рисунок-3.4 – Диаграмма классов

Классы диаграммы классов:

- LoginUser – авторизация пользователей;
- RegisterUser – регистрация новых пользователей;
- ProfileView – создание профиля пользование, редактирование, представление информации о пользователе;
- ContactCreate – представление функции обратной связи;
- TaskCreate – создание задач и заказов;
- TaskSearch – пойск задач по БД на сайте;
- статический класс TaskView – представляет информацию о задачах на странице;
- статический класс TaskCatView – представляет информацию о категориях задач и заказов;
- MessengeDirect – возможность пользователям чатится между друг другом.

3.3.3 Диаграмма последовательности

Как было уже сказано Use case диаграмма показывает сценарий пользователя на сайте, что же нужно, чтобы показать подробные действия посетителя на сайте. Диаграмма последовательности показывает сценарий взаимодействия между актером и объектом использования. Построение представляет собой временные объекты, которые начинаются сверху и по не многу идут вниз, чтобы показать последовательность взаимодействия от начала использования до последнего сообщения от сервера.

Примеры использования методов описанные на диаграммах последовательности предсталенны ниже следующим образом:

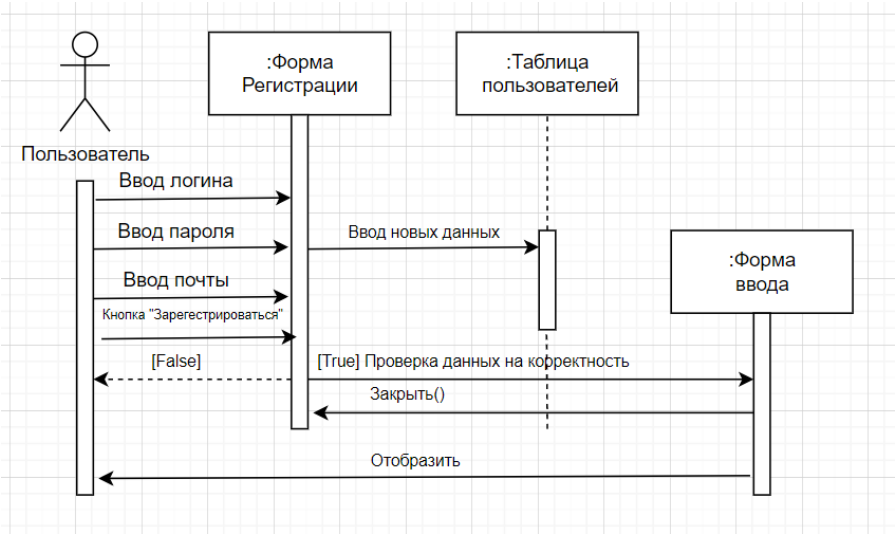

Рисунок-3.5 – Диаграмма последовательности «Регистрации на портал»

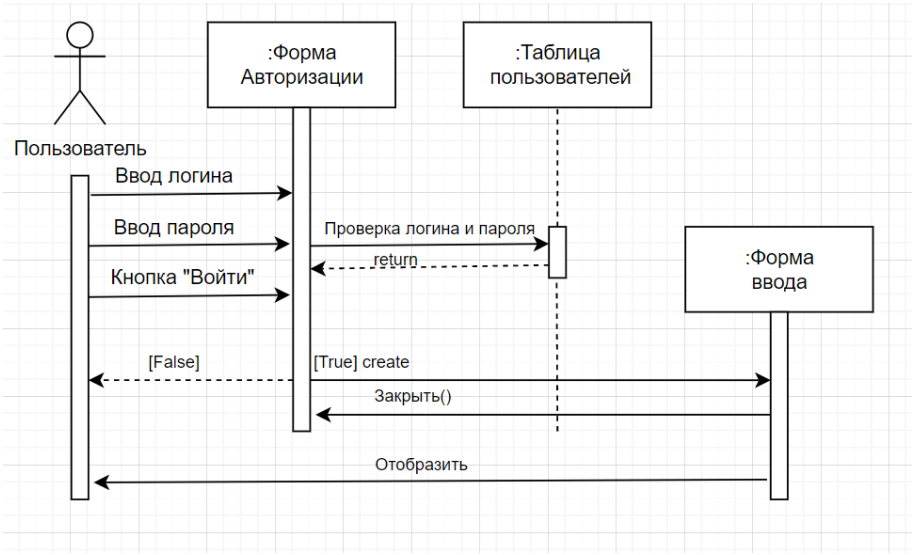

Рисунок-3.6 – Диаграмма последовательности «Авторизации на портал»

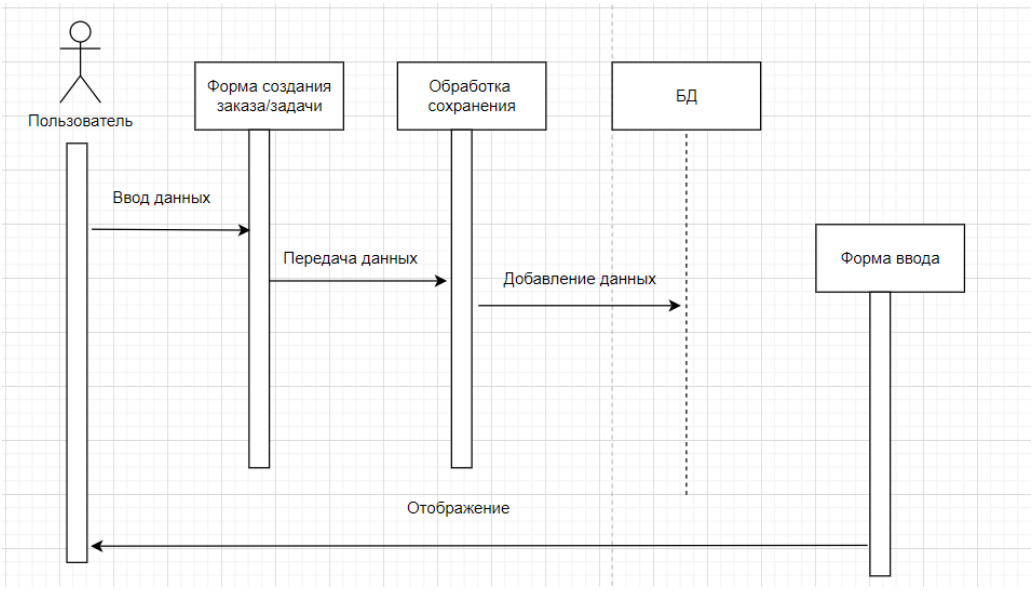

Рисунок-3.7 - Диаграмма последовательности «Создание заказа/задачи»

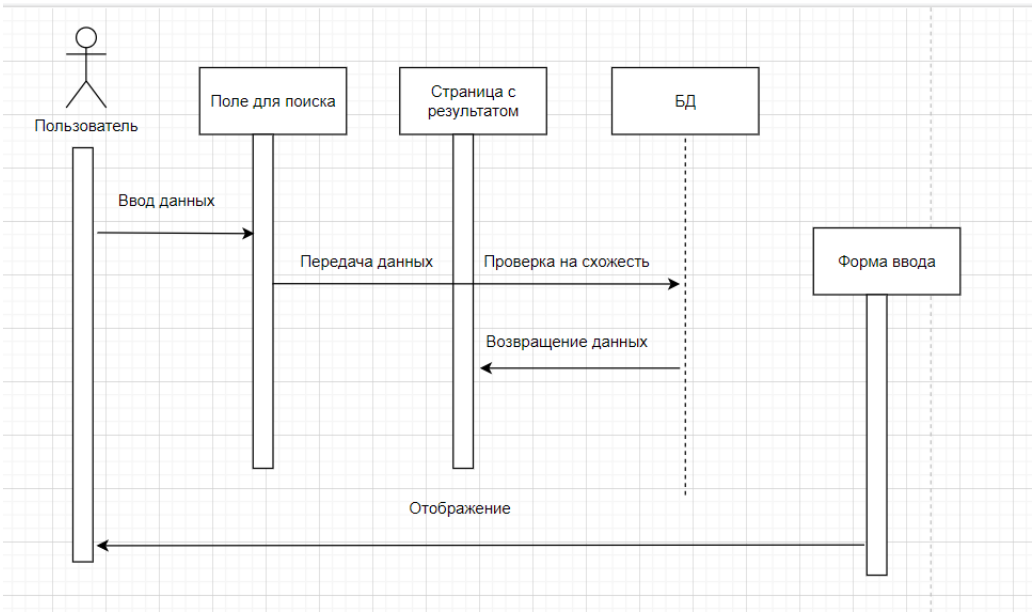

Рисунок-3.8 – Диаграмма последовательности «Навигация по сайту»

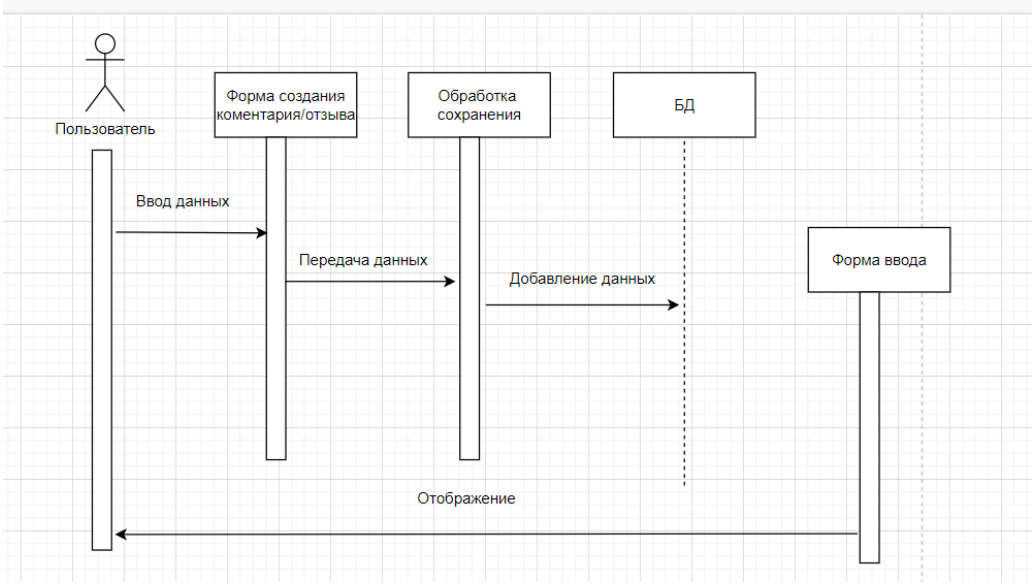

Рисунок-3.7 – Диаграмма последовательности «Создание комментария»

4 Разработка системы

Разработка системы самая важная часть дипломного проекта и для ее успешного выполнения нужно сделать следующие:

- Выбрать правильное программные обеспечения для разработки сайта, основываясь на новых разработках, взвесим все плюсы и минусы;
- Успешно воссоздать примерный интерфейс взаимодействия системы для пользователя (Front-end), используя графические редакторы;
- Реализовать функциональные возможности сайта, описанные в диаграммах и прописать логистику системы (Back-end), используя языки программирования, фреймворки и остальные расширения;
- Осуществить оптимизацию или ускорения системы;
- Создать и настроить панель администратора, для размещения и слежкой за контентом, просмотра действий пользователей;
- Выбрать подходящий хостинг и развернуть сайт на нем, выбрать доменное имя.

4.1 Описание программного обеспечения

Разработка своего портала фрилансеров, уже говорит о том, что для реализации проекта нужно правильное программное обеспечение, для создания системы. Для начала ПО, которые будут использоваться в создании проекта, а именно:

- среда разработки системы;
- языки разметки страницы, для описания шаблонов сайта;
- языки программирования;
- Framework и другие вспомогательные расширения;
- хостинг для сайта.

4.1.1 Среда разработки

Каждый разработчик прежде, чем собирается писать сайт знает, что написать код можно и в обычном текстовом редакторе, но зачем так сильно усложнять себе жизнь, для этого были разработаны среды разработки (IDE Integrated Development Environment), которые являются отлично интегрированными платформами для создания разного ПО. Среда разработки является совокупностью большого количества инструментов, таких как:

- текстовый редактор;
- компилятор;

- терминалы;
- средства сборки и отладчик.

На данный момент существуют множество отличных IDE, которые отличаются только в направлении языков программирования. Некоторые IDE являются платными, поддерживают узкий круг языков программирования и могут быть совместимы не со всеми операционными системами.

Для своего проекта, мною был выбран Visual Studio Code, который поддерживался моей операционной системой Windows и являлся отличным ПО, в котором я про работал за время прохождения учебы в университете. Отличными и положительными качествами VS Code можно называть следующее:

- огромный спектр поддерживаемых языков программирования;
- совершенно бесплатное ПО;
- совместимость сразу с Windows/Linux/MacOS;
- дает возможность изменять рабочую панель;
- имеет возможность установить огромное количество расширений и плагинов для работы с кодом;
- является легким и удобным редактором, поэтому применяется для решения быстрых задач;
- простая работа с пакетными менеджерами.

Как итог, действительно существует множество сред для разработки ПО, такие как PyCharm, Visual Studio, WebStorm или IntelliJ IDEA, но отличий среди них очень мало и интерфейс у всех схож и по своему опыту, лучше выбрать то, что больше знакомо и не будет занимать время на установку и обновления.

4.1.2 Языки разметки страницы. Формальный язык программирования

Web разработка это первым делом умение правильно описать интернет-документ, то есть свой сайт, для этого я выбрал язык разметки страницы HTML (Hyper Text Markup Language). Существуют так же множество остальных языков разметки такие как: XML, Wiki, BBCode, Textile, но так как HTML является наиболее популярным и распространённым языком среди всех остальных. Язык HTML является уже стандартом для разработчиков и интернет-документов, благодаря ему и создаются web-страницы на сайтах. Документ содержащий в себе код языка HTML отображают страницы в браузере, показывая уже статический и динамический контент. Данный язык может содержать в себе множество элементов языков программирования, к примеру для описания стилей

элементов страниц, а также для установки динамических и статических данных.

Портал «TaskStart» в общем имеет 21 web-страницы написанные на html:

- index – главная страница сайта;
- login страница с формой авторизации;
- register страница с формой регистрации;
- contact_us – страница с формой обратной связи;
- about_us – страница с информацией о сайте;
- craft – страница, содержащая форму создания заказов и задач;
- podrubrika – страница отображает информацию о существующих заказах и задачах, выбранных по конкретной категории;
- design – страница отображает существующие категории по рубрике «Дизайн»;
- development - страница отображает существующие категории по рубрике «Разработка и IT»;
- audio_video - страница отображает существующие категории по рубрике «Аудио, видео, съемка»;
- seo - страница отображает существующие категории по рубрике «SEO»;
- promotion - страница отображает существующие категории по рубрике «Социальная сеть и реклама»;
- user – страница профиля пользователя;
- edit_profile – страница с формой редактирования профиля;
- task_info – страница с информацией о задаче или заказе;
- edit_task – страница с редактированием заказа или задачи;
- comment_add – страница с формой создания комментариев под любыми задачами и заказами;
- search_results – страница, содержащая информацию о поиске;
- base – навигационная панель для чата;
- direct – страница с чатом между пользователями, отображает людей, с которыми идет или шел диалог и позволяет увидеть текст переписки;
- search_user - страница, содержащая информацию о поиске пользователей;

Дальше рассмотрим формальный язык программирования. CSS – это и есть формальный язык программирования, роль которого задать и описать с помощью классов внешний вид страницы. Если язык разметки был использован HTML, то благодаря этому, есть возможность изобразить цвета, шрифты, разные положения объектов, изменения размеров изображения и так далее.

Ради примера можно рассмотреть одну из созданных мною страницы написанной на HTML с подключенной к нему стилем CSS:

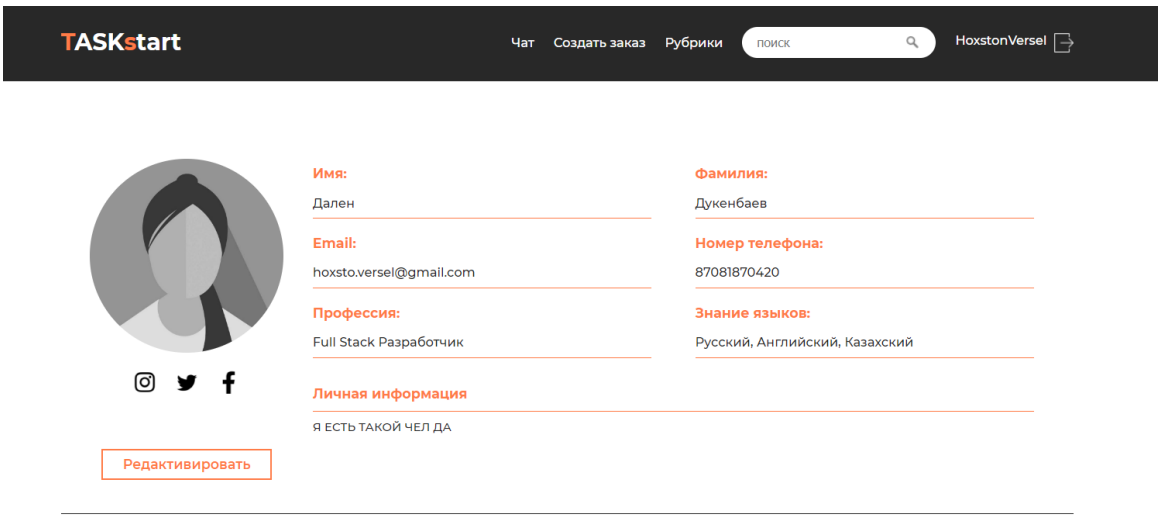

Рисунок-4.1 – Профиль пользователя «TaskStart»

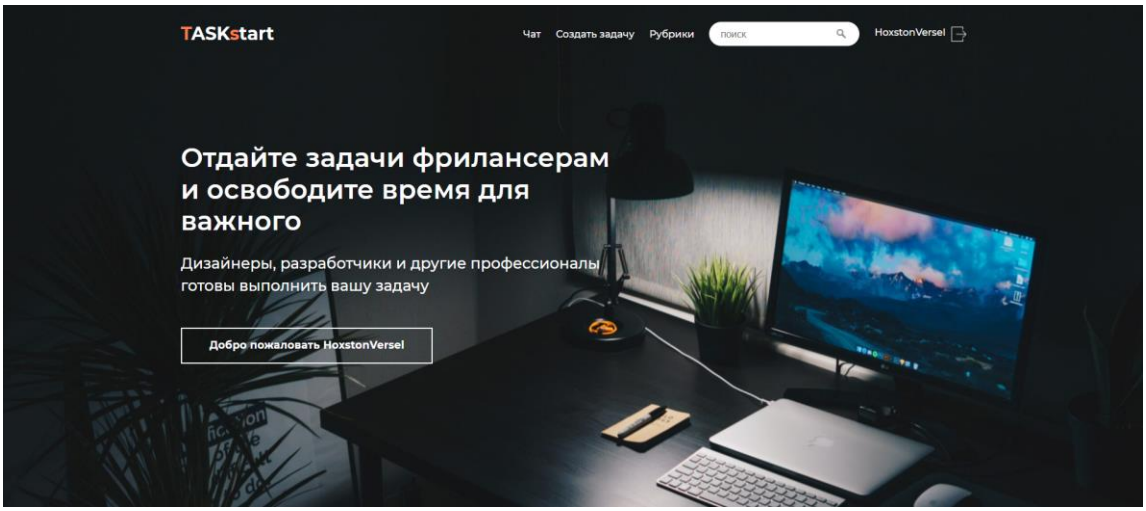

Рисунок-4.2 – Главная страница сайта «TaskStart»

Как видно на шаблоне главной странице был размещен статический и динамический контент, такие как:

- навигационная панель;
- задний фон;
- надписи и их шрифты;
- логотип;
- динамическое имя пользователя;
- кнопки.

На шаблоне страницы профиля размещена информация о пользователе, аватар, иконки социальных сетей пользователя и кнопка редактирования.

4.1.3 Язык программирование. Framework Django

Любое приложение, которое создает разработчик, обращается к нему на своем языке, и мы свойственно должны тоже уметь использовать тот или иной язык программирования, чтобы машина могла нас понимать. Для своего проекта, я выбрал язык программирования Python.

Python очень универсальный язык программирования, он высоко оценен в разных сферах деятельности, к примеру для системных администраторов, Python используется в целях автоматизации управления и слежения за данными, которые с помощью специальный библиотек они могут отобразить информацию либо в 2D, либо в 3D графиках. Для web-разработки данный язык, обычно и выбирают разработчики, но также связано это с тем, что Python хорошо себя показывает в решениях с машинным обучением и использованием его в разработке мобильных приложений. Обычно в веб-разработке используется специальный framework и самый популярный из них это «Django».

Framework Django – является известным фреймворком на языке программирования Python, так же является очень мощным и эффективным, в силах которого сразу дать пользователю начать работу. Использующие его разработчики, называют его простейшим, но это не значит, что на нем нельзя писать сложные web-приложение, совсем наоборот, к примеру интернет-платформа «YouTube» полностью написана на языке программирования «Python», тем самым на фреймворке Django.

Я выбрал Django, потому что тот имеет в себе все самое необходимое для написания простого и чистого кода, куча полезных библиотек, хорошая документация, имеет встроенную панель администратора и отлично работает с реляционными базами данных.

Так же можно было бы выбрать язык программирования JavaScript, который предназначался для веб разработки и использовать его фреймворк Node.js, но написания дипломного проекта на новом языке и не знакомом фреймворке, приносит больше пользы и намного больше опыта в работе с разными разработками.

4.2 Интерфейс системы

Как и любой другой сайт, портал фрилансеров должен иметь свой оригинальный пользовательский интерфейс, веб-интерфейс. Благодаря интерфейсу на сайте, пользователь будет лучше ориентироваться по сайту, так как это совокупность связанных между собой страниц и методов. Веб-страница — это только часть интерфейса и является лишь холстом для выбора места размещения интерфейса.

Первым что всегда бросается в глаза любого пользователя это цветовая палитра сайта. Цвет всегда влияет на восприятие человека о сайте, цвета могут говорить людям об опасности, заинтересовать человека, заинтриговать и успокоить, так же и я, выбирав цвета для сайта был полностью уверен, что должен собрать хорошую цветовую гамму.

Цветовая гамма сайта подобрана специально чтобы сосредоточить пользователя на своих основных целях, а именно:

- найти клиента;
- найти фрилансера;
- завершить заказ или задачу.

В ходе моего исследования, опроса среди коллег и друзей были выбраны следующие цвета:

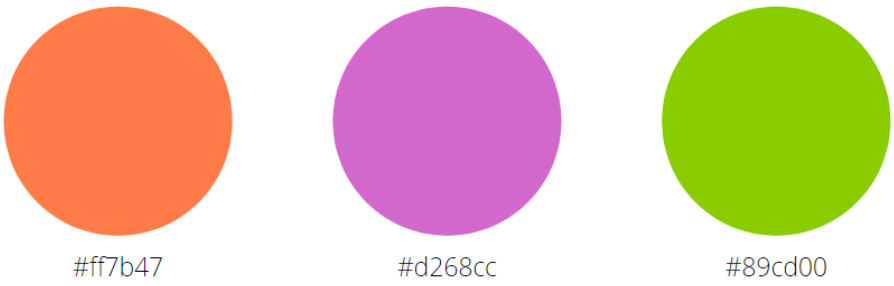

Рисунок-4.3 – Цветовая гамма сайта

Графический интерфейс на сайте будет являться для пользователя методами ввода данных, а система будет изображать эти данные на экране, к примеру это должно выглядеть так:

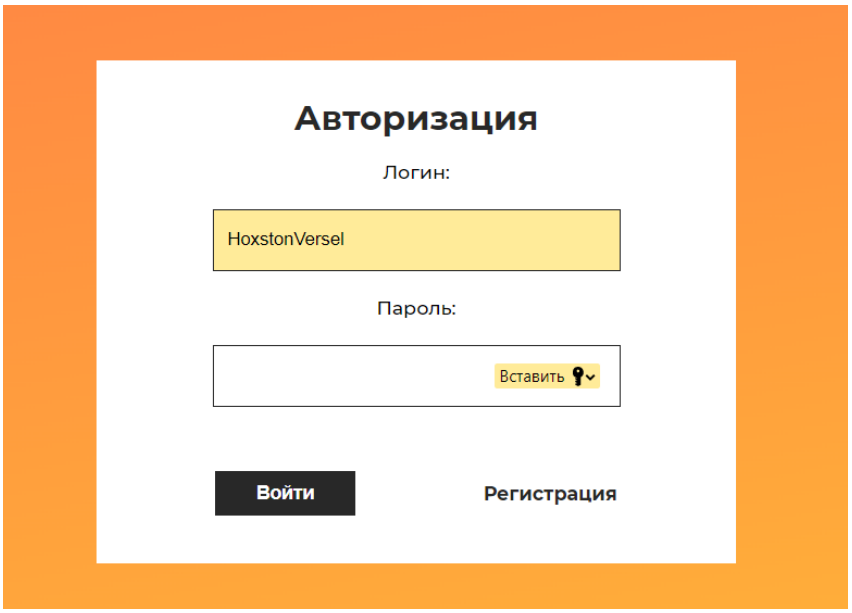

Рисунок-4.4 – Графический интерфейс «Авторизация»

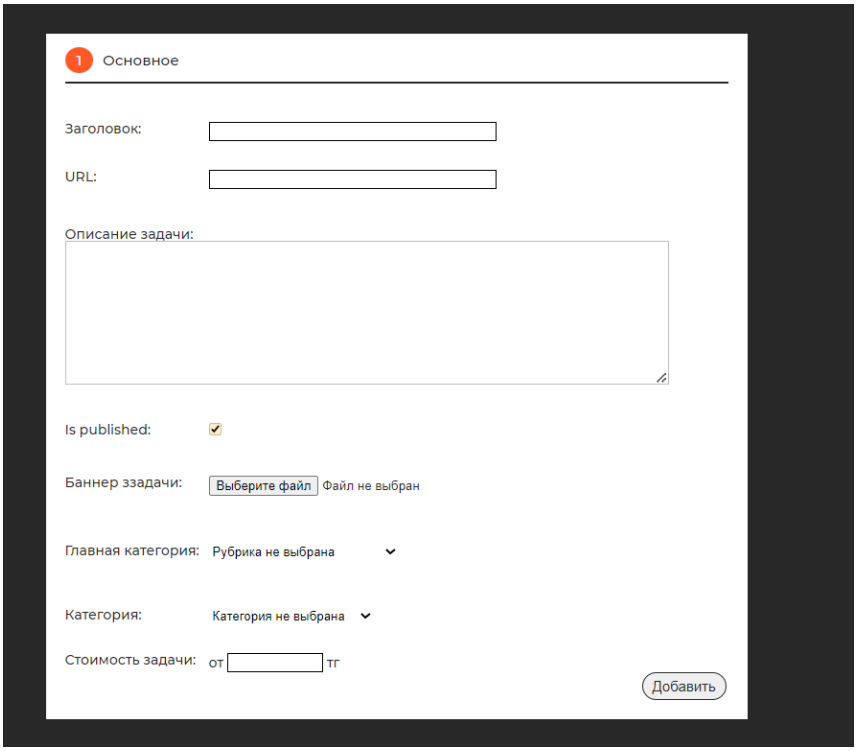

Рисунок-4.5 – Графический интерфейс «Создание заказа/задач»

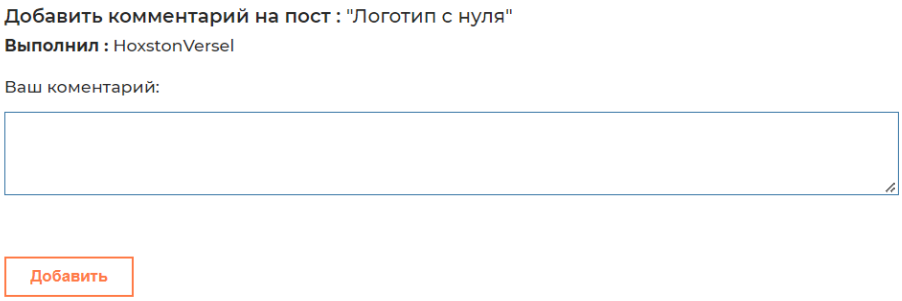

Рисунок-4.6 - Графический интерфейс «Написание комментария»

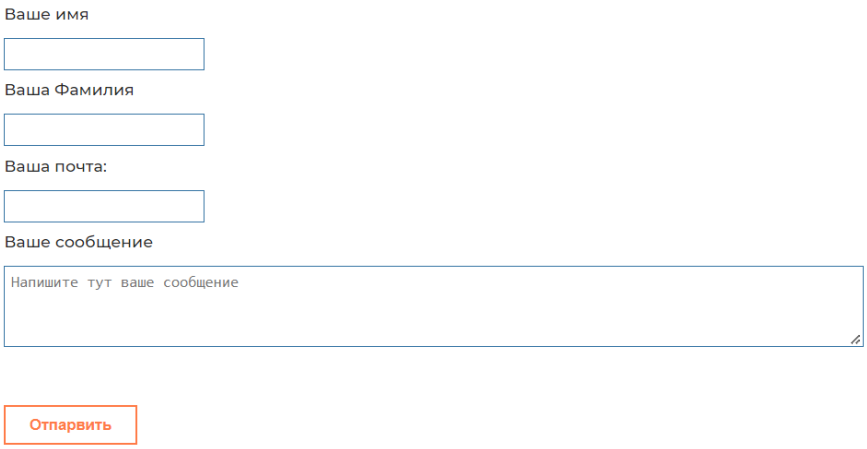

Рисунок-4.7 - Графический интерфейс «Обратная связь»

Каждый сайт имеет свой собственный, может и не уникальный, но адаптивную панель навигации для пользователей:

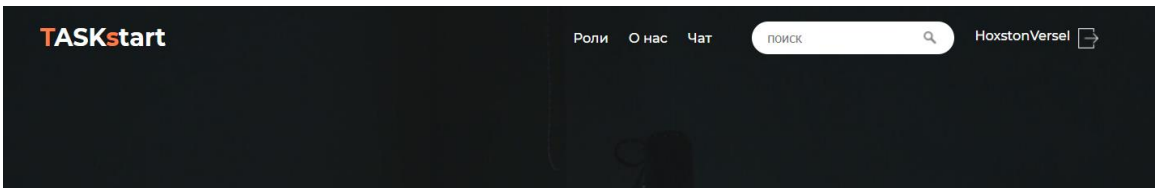

Рисунок-4.8 – Адаптивное меню навигации (PC)

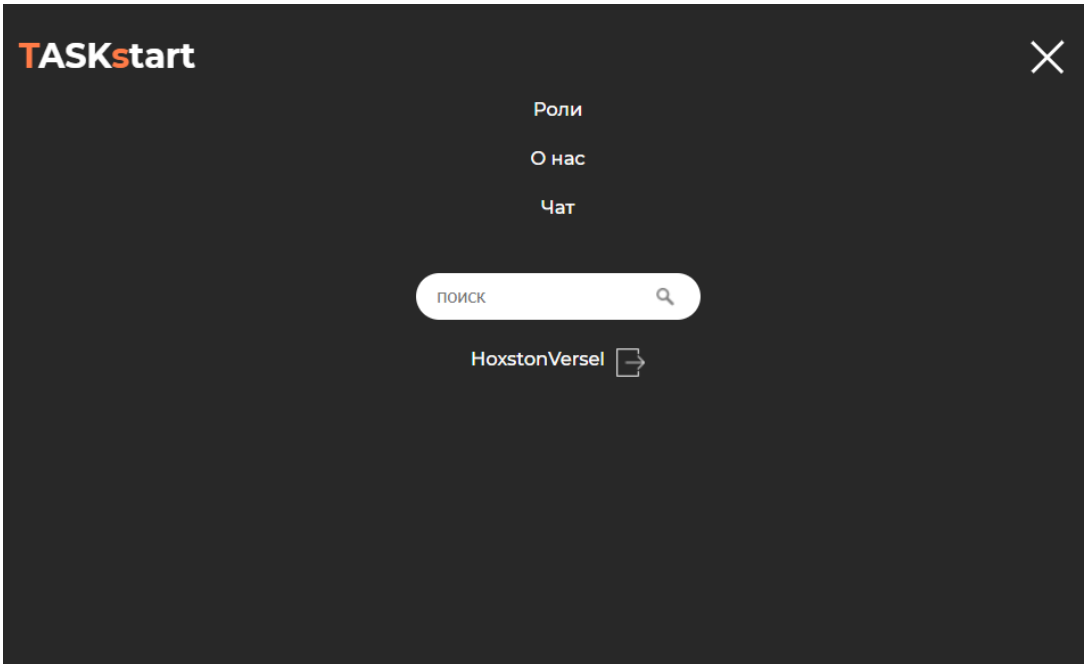

Рисунок-4.9 – Адаптивное меню навигации (Мобильная версия)

4.3 Функционал системы

Основным функционалом системы это методы создания задач и заказов для фрилансеров и клиентов, в тоже время обязательно на портале должен существовать чат, личные сообщения пользователей, чтобы те в свою очередь могли договариваться о той или иной задаче, ввести переговоры о цене и заказе в целом.

Функционал системы строиться на проектирования моделей базы данных, так же создание классов и функций. Модели базы данных Django являются MTV или простыми словами толстые модели. Являются они таковыми по причине их простоты и в эффективной работе в разработке приложений.

```
20
21  class Task(models.Model):
22      title        = models.CharField(max_length=255 , verbose_name = 'Заголовок')
23      slug         = models.SlugField(max_length=255, unique=True, db_index=True, verbose_name="URL")
24      content      = models.TextField(blank=True, verbose_name = 'Описание задачи')
25      photo        = models.ImageField(upload_to="images/%Y/%m/%d/", verbose_name = 'Баннер ззадачи')
26      time_create  = models.DateTimeField(auto_now_add=True, verbose_name = 'Дата Создания')
27      time_update  = models.DateTimeField(auto_now=True, verbose_name = 'Дата обновления')
28      is_published = models.BooleanField(default=True)
29      price        = models.CharField(max_length=20, verbose_name = 'Стоимость задачи')
30      main_cat     = models.ForeignKey('Main_Category', on_delete=models.PROTECT, null = True, verbose_name = 'Главная категория')
31      cat          = models.ForeignKey('Category', on_delete=models.PROTECT, verbose_name = 'Категория')
32      user         = models.ForeignKey(User, on_delete=models.CASCADE, blank=True)
33
34      def __str__(self):
35          """Return title and username."""
36          return '{} by @{}'.format(self.title, self.user.username)
37
38      def get_absolute_url(self):
39          return reverse('post', kwargs={'post_slug': self.slug})
40
41
42      class Meta:
43          verbose_name = 'Задача'
44          verbose_name_plural = 'Задачи'
45          ordering = ['time_create', 'title']
46
```

Рисунок-4.10 – Модель базы данных (Задачи пользователя)

Django славится своей встроенной базой данных и как систему управление базами данных, мою было выбрано не сильно требовательное приложение «SQLite Studio».

Дальше идет шаг в разработке своих первых форм и написание классов для отправления запросов на сервер и получения данных от сервера:

```
162 # ---Добавить заказ на сайт---
163 def add_post_view(request):
164
165     if request.method == 'POST':
166         form = AddPostForm(request.POST, request.FILES)
167         if form.is_valid():
168             task = form.save(commit=False)
169             task.user = request.user
170             task.save()
171             return redirect('home')
172     else:
173         form = AddPostForm()
174
175     return render(request, 'freelance/craft.html', {'title': 'Добавление заказа', 'form': form})
176 # ---Добавить заказ на сайт---
```

Рисунок-4.11 – Функция добавления задачи на сайт

Процесс создания на сайте проходит, после заполнения формы пользователем, далее отправляется POST запрос на сервер для сохранения этих данных на сервере, как и показано диаграмме последовательности.

4.4 Оптимизация системы

Оптимизация сайта, это процесс уменьшение нагрузки на сам сайт, при частых потоков данных, обработки данных и отправки с сервера к пользователю. Как часто бывает, хороший и надежный хостинг, берет на себя часть нагрузки, но несмотря на это на сайте должно быть рассмотрены следующие методы оптимизации системы:

- скорость работы самого приложения;
- конкретные возвращаемые данные;
- нагрузка на базу данных.

Так и для отслеживания правильно ли система работает, лучше воспользоваться встроенным в фреймворк инструментом «Django Debug Toolbar». Установив этот инструмент, на странице сайта, а именно в панели разработчика, мы будем видеть следующее:

- версию Django;
- время формирования страниц;
- количество выполненных SQL запросов;
- список задействованных шаблонов.

Благодаря инструменту, мы можем более подробно развернуть каждый пункт и проверить полную информацию. Как и с SQL запросами, нужно уменьшить время на про грузку страниц и контента, для этого используем механизм кэширования страниц, который почти всегда используется на каждом сайте.

Кэширования в браузере играет роль моста между пользователем и страницей, который уменьшает нагрузку на сеть пользователя, сохраняя уже прогрузившиеся элементы страницы в памяти. Отсюда следует, что при частом появлении на сайте, пользователь может словить кучу ошибок, ведь на сайтах всегда происходят тех обслуживания, добавление новых данных, элементов, обновление скриптов и стилей. Как правило все обновляется на сервере, когда со стороны клиента ничего не меняется, лишь потому что, пользователь не обнулил свой кэш. Чтобы не просить пользователя заниматься этим, чаще всего данный процесс автоматизируют, что в моем случае, я могу реализовать кэширование для следующих элементов:

- база данных;
- представлений;
- шаблонов.

4.5 Настройка админ панели

Django admin panel является встроенным приложением фреймворка Django, который исполняет роль помощника для модерирования сайта, а именно:

- просмотр созданных записей в БД;
- обновление контента сайта;
- удаление записей.

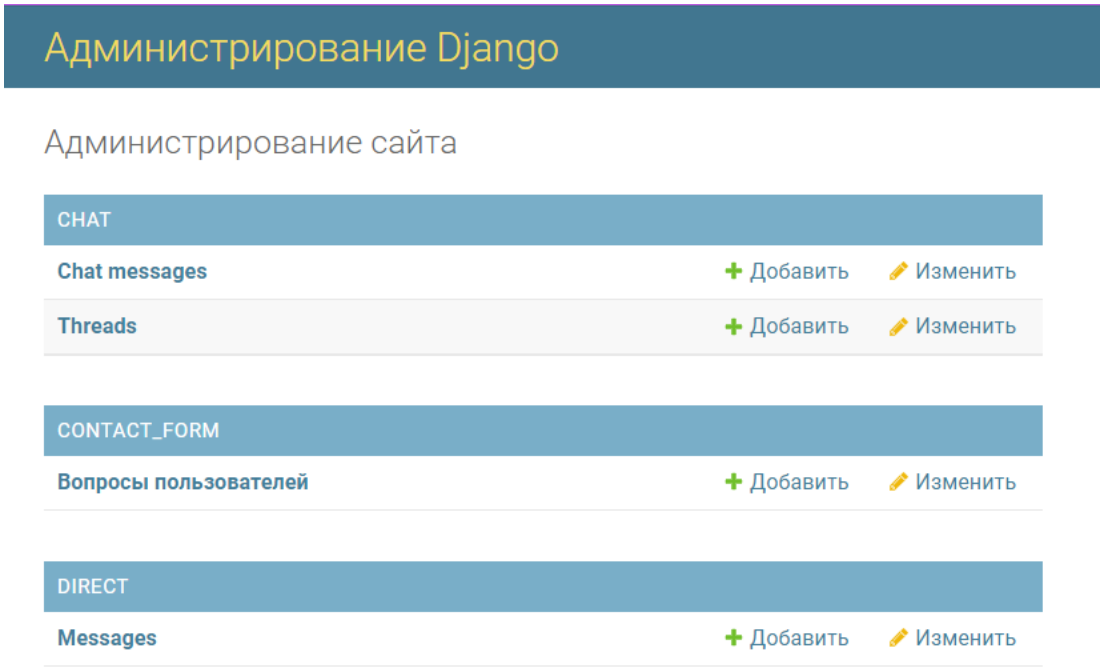

Рисунок-4.12 – Просмотр данных из админа панели

Так же можно подробно изучить те данные, которые находятся в этих объектах. Например «Вопросы пользователей» носят в себе информацию обратной связи:

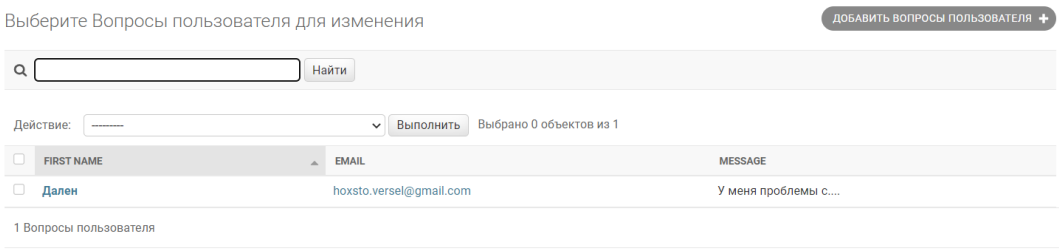

Рисунок-4.13 – Просмотр существующих записей через админа панель

Теперь глянем, как мы можем более углубленно проверить данную информацию, изменить или удалить ее:

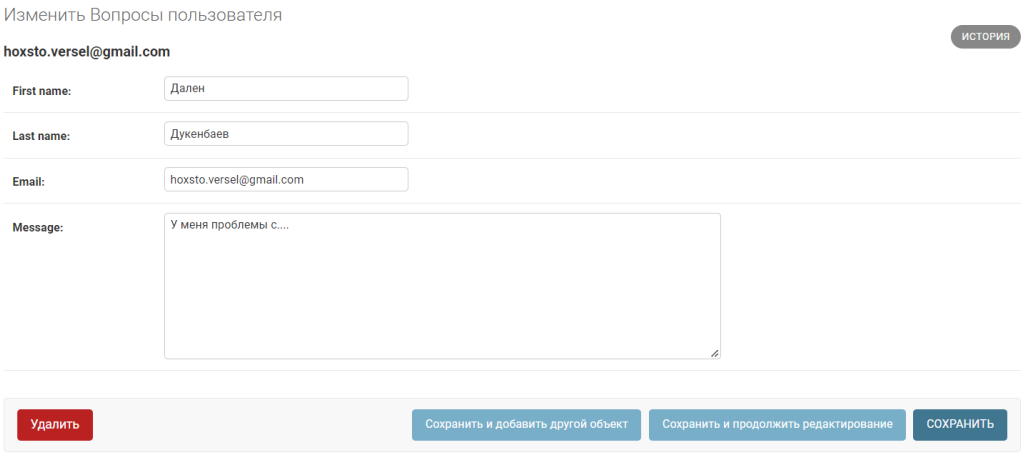

Рисунок-4.14 – Подробный просмотр записи через админа панель

4.6 Развертывание сайта на хостинге

Развертывания сайта на хостинге будет происходит в два этапа. Первый этап — это выбор надежного хостинга, далее создания доменного имени и установка нужных компонентов, а именно нужной версии Python и фреймворка Django и всех его директорий. Второй этап заключается во вложении всех компонентов нашего портала и связывания его с директорией хостинга.

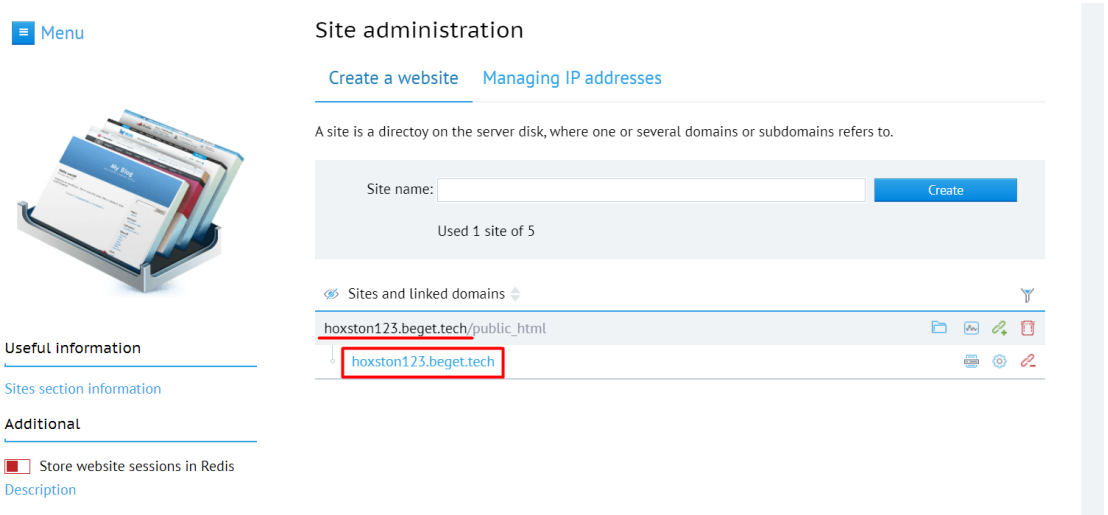

Рисунок-4.15 – Выбор доменного имени и настройка сайта на хостинге

4.7 Тестирование

Разработка портала включает в себя множество форм, классов, методов и моделей базы данных, за достоверной работой которых, нужно правильно следить и тестировать процесс их выполнения. Далее в ходе разработки добавление даже минимального изменение, может повлечь за собой ряд багов и ошибок, отсюда следует что тестирование системы, играет наиважнейшую роль при создании любого продукта.

Я считаю, что тестирование можно разделить на два вида ручное и внутри кодовое или, иными словами, автоматизированное тестирование вашего приложения. Лучше всего, я думаю, будет использовать ручное тестирования сайта, но только в том случае, если проект уже находится на достаточно длительном времени разработки. Когда же при начальной разработке мы можем автоматически тестировать нашу систему, даже при малейших изменениях, что хорошо скажется на процессе создания портала.

В совокупности, тесты делятся на типы, а именно:

- юнит-тесты;
- интеграционные тесты;
- регрессионное тестирование.

Для наглядности работы с тестами, лучше использовать методы юнит-тестов, которые будут проверять компоненты отдельно, так как в случае других типов, использовать их можно уже только имея готовый проект и имея за своей спиной опыт работы с ними. Так что именно мы будем тестить в системе, что на это говорит фреймворк Django? В Django существует директория, которая отлично подходит для написания тестов, которая будет соответствующе имитировать URL запросы, вводить тестовые данные и анализировать выходные данные. Единственное что мы, пожалуй, не будем тестировать это добавленные библиотеки, то есть все что предоставляет нам Django. После любого теста, мы можем наблюдать за результатами тестирования в терминале нашей среды разработки:

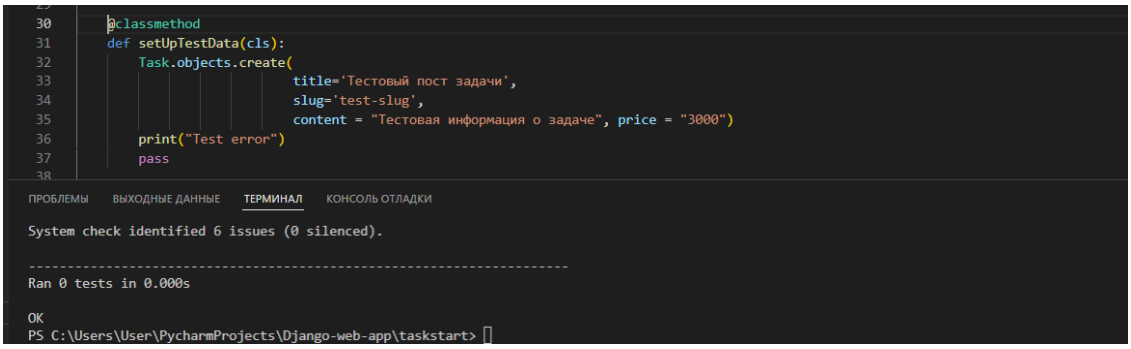

Рисунок-4.16 – Результат тестирования простого юнит-теста

ЗАКЛЮЧЕНИЕ

При выполнении дипломного проекта и разработки веб-приложения, мною было совершенно множество успешно выполненных пунктов и подпунктов к требованию к системе. В настоящее время сайты поиску удаленной работы или работы, как фрилансер в целом, очень мало, но мой проект должен формировать у людей понятие о бирже фриланса и привлечь большую аудиторию развивать данную деятельность в Казахстане.

Были проделаны работы по следующим основным этапам разработки портала:

- изучение рынка, биржи фриланса;
- определение требований системы;
- проектирование архитектуры системы и модели UML;
- разработка приложения, а именно его функционал и интерфейс.

Изучение рынка включало в себя цель определить причину спада фриланса в Казахстане, проанализировать ситуацию с удаленной работой и сделать по этому поводу выводы.

Определить требования к системе — это четкое представление о том, как должен выглядеть проект, какие функции он должен уметь, на сколько он должен быть быстром и на сколько можно безопасным.

Портал фрилансеров- должен не просто служить людям для взаимовыгодных отношений и поиску специалистов для их задачи, портал должен привлечь к себе молодую аудиторию, которая будет развиваться в сфере IT и развивать наше IT.

Если принять во внимание функциональные требования, то сайт может:

- регистрация и авторизация пользователя на сайт;
- создания своего профиля на сайте;
- переключаться между ролями, а именно фрилансером и клиентом;
- возможность показывать выставленную информацию о задаче или о заказе;
- создание задач и заказов на сайте;
- комментирование задач или заказов;
- метод общения через интернет, договариваться о цене и заказах;
- функция для обратной связи с пользователями.

На разработку интерфейса системы, было потрачено больше всего времени и если ссылаться на более облегченную разработку тех же задач, то лучше всего выбрать какой ни будь популярный Фреймворк для быстрого дизайна и размещения элементов страницы по местам.

Как итог сайт состоит и большого количества шаблонов, связанных между собой, в которых лежат разный статический и динамический контент. Так же успешно была произведена адаптация к интерфейсу сайта, для мобильных версий браузера.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1 Анализ спроса и предложения фрилансеров на примере биржи oDesk // Электронная версия на сайте https://habr.com/ru/post/240979/.

2 Прогнозы биржи фриланса 2021-2022 год // Электронная версия на сайте https://blog.kwork.ru/rynok-frilansa/frilans-2021-itogi-goda-i-prognoz-na-2022.

3 Фриланс в Казахстане // Электронная версия на сайте https://edunet.pro/freelancers-tribune/frilans-v-kazahstane-sajty-rabota-obucenie.

4 Написание технического задания // Электронная версия на сайте https://texterra.ru/blog/kak-sostavit-gramotnoe-tekhzadanie-na-razrabotku-sayta.html.

5 Айдас Бендорайтис, Джейк Хроника. Django 3 Web Development Cookbook // Электронная версия на сайте https://vk.com/wall-54530371_310677.

6 UML моделирование // Электронная версия на сайте https://habr.com/ru/post/458680/.

7 Язык разметки HTML // Электронная версия на сайте https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/HTML_basics.

8 Язык стилей CSS // Электронная версия на сайте https://skillbox.ru/media/code/chto_takoe_css/.

9 Язык программирования Python // Электронная версия на сайте https://all-python.ru/osnovy/yazyk-programmirovaniya.html.

10 Framework Django // Электронная версия на сайте https://proproprogs.ru/django.

11 Интерфейс и взаимодействия человека с сайтом // Электронная версия на сайте https://sales-generator.ru/blog/interfeys-sayta/.

12 Модели базы данных Django // Электронная версия на сайте https://habr.com/ru/post/87357/.

13 Функции и классы Django // Электронная версия на сайте https://www.djbook.ru/rel1.9/topics/class-based-views/intro.html.

14 Оптимизация сайта // Электронная версия на сайте https://proproprogs.ru/django/optimizaciya-sayta-s-django-debug-toolbar.

15 Кэширование системы // Электронная версия на сайте https://proproprogs.ru/django/django-vklyuchaem-keshirovanie-dannyh.

16 Панель администратора // Электронная версия на сайте https://proproprogs.ru/django/nachinaem-rabotu-s-admin-panelyu.

17 Написание тестов на фреймворке Django // Электронная версия на сайте https://developer.mozilla.org/ru/docs/Learn/Server-side/Django/Testing.

18 Антоние Меле. Django 3 for example// Электронная версия на сайте https://djangobyexample.com/.

19 Пол МаКфедрис. Web Design Playground (2019)// Электронная версия на сайте https://vk.com/wall-54530371_296845.

20 Учебник HTML с CSS// Электронная версия на сайте https://www.schoolsw3.com/html/html_css.php.

21 Основы JavaScript. // Электронная версия на сайте https://learn.javascript.ru/first-steps.

Приложение А
(обязательное)

Техническое задание

А.1.5 Техническое задание на разработку портала фрилансеров

А.1.5.1 Назначение

Система предназначена для фрилансеров, сохранения их личного времени, работая на себя. Данное веб-приложение служит для создания задач фрилансерами, которые тем самым будут искать потенциальных клиентов, где портал выступит в роли посредника между специалистом и клиентом.

А.1.5.2 Требования к функциональным характеристикам

Система должна обеспечивать возможность выполнения следующих функций:

- регистрация новых пользователей;
- авторизация существующих пользователей;
- сброс пароля и смена его на новый;
- изменение логина и почты пользователя;
- редактирования профиля пользователя;
- создания заказов для фрилансеров;
- редактирование заказа, написание новой или дополнительной информации к существующей;
- удаление существующего заказа;
- создание задач для потенциальных заказчиков;
- редактирование задач, написание новой или дополнительной информации к существующей;
- удаление существующих задач;
- писать отзывы к существующим заказам и задача;
- писать отзывы на странице профиля другого пользователя;
- создания формы запроса обратной связи и помощи;

А.1.5.3 Требования к надежности

Создание панели администратор, добавление модерации, слежка за действиями пользователей, блокировка пользователей с условием их некорректной деятельности на портале. Так же на панели администратора должно быть предусмотрено просмотр созданной информации и заполнение контента информацией.

А.1.5.4 Требования к составу и параметрам технических средств

Так как система является веб приложениям, то создаваемый продукт должен быть совместим для браузеров всех типов, на персональных компьютерах пользователей, а так же на мобильных устройствах.

А.1.5.5 Требования к информационной и программной совместимости

Система должна быть совместима со всеми операционными системами, такими как Windows, Linux и MacOS, на всех поддерживаемых ими браузерами.

А.1.5.6 Требования к программной документации

Разрабатываемые программные модули должны быть самодокументированы, т.е. тексты программ должны содержать все необходимые комментарии.

Программная система должна включать справочную информацию о работе и подсказки пользователю при взаимодействии с пользовательским интерфейсом, а выполненные методы на сайте должны предупреждать пользователя о удачно или не удачном совершении какой-либо отправки данных на портал.

Приложение Б

Логика и текст ПО TaskStart

```
def index(request):

    user_info = Profile.objects.all()
    paginator = Paginator(user_info, 5)

    user_list = request.GET.get('user_list')
    page_obj = paginator.get_page(user_list)
    context = {


        'title': 'Главная страница',
        'user_info': page_obj,
    }
    return render(request, 'freelance/index.html', context=context)

def freelancer(request):
    context = {
        'title': 'Страница фрилансеров',
    }
    return render(request, 'freelance/freelancer.html', context=context)

def client(request):
    context = {
        'title': 'Страница клиентов',
    }
    return render(request, 'freelance/klient.html', context=context)

# Главные рубрики заказов
def show_main_cat(request):
    return render(request, 'freelance/rubrica.html')

# Категории заказов
def categories(request):
    return render(request, 'freelance/categories.html', {'title': 'Категории'})

def about_is_info(request):
    context = {
        'title': 'Главная страница',
    }
    return render(request, 'freelance/about_us.html', context=context)
```

Продолжение приложения Б

```
#Страница не найдена
def pageNotFound(request, exception):
    return HttpResponseNotFound('<h1>Страница не найдена</h1>')

# Рубрика: Дизайн
def design(request):
    cats = Category.objects.all()
    paginator = Paginator(cats, 10)
    cat_list = request.GET.get('cat_list')
    page_obj = paginator.get_page(cat_list)

    context = {
        'page_obj': page_obj,
        'cats': cats,
        'title': 'Дизайн',
        'cat_selected': 0,
    }
    return render(request, 'freelance/design.html', context=context)

# Рубрика: Разработка и IT
def development(request):
    cats = Category.objects.all()
    paginator = Paginator(cats, 20)
    cat_list = request.GET.get('cat_list')
    page_obj = paginator.get_page(cat_list)

    context = {
        'page_obj': page_obj,
        'cats': cats,
        'title': 'Разработка и IT',
        'cat_selected': 0,
    }
    return render(request, 'freelance/development.html', context=context)

# Рубрика: Социальная сеть
def promotion(request):
    cats = Category.objects.all()
    paginator = Paginator(cats, 30)
    cat_list = request.GET.get('cat_list')
    page_obj = paginator.get_page(cat_list)

    context = {
```

```
        'page_obj': page_obj,
        'cats': cats,
        'title': 'Социальная сеть',
        'cat_selected': 0,
    }
    return render(request, 'freelance/promotion.html', context=context)

# Рубрика: SEO
def seo(request):
    cats = Category.objects.all()
    paginator = Paginator(cats, 37)
    cat_list = request.GET.get('cat_list')
    page_obj = paginator.get_page(cat_list)

    context = {
        'page_obj': page_obj,
        'cats': cats,
        'title': 'SEO',
        'cat_selected': 0,
    }
    return render(request, 'freelance/seo.html', context=context)

# Рубрика: Аудио, видео, съемка
def audio_video(request):
    cats = Category.objects.all()
    paginator = Paginator(cats, 50)
    cat_list = request.GET.get('cat_list')
    page_obj = paginator.get_page(cat_list)

    context = {
        'page_obj': page_obj,
        'cats': cats,
        'title': 'Аудио, видео, съемка',
        'cat_selected': 0,
    }
    return render(request, 'freelance/audio_video.html', context=context)

# ---Добавить задачи на сайт---
def add_post_view(request):

    if request.method == 'POST':
        form = AddPostForm(request.POST, request.FILES)
```

```
            if form.is_valid():
                task = form.save(commit=False)
                task.user = request.user
                task.save()
                return redirect('home')
        else:
            form = AddPostForm()

        return render(request, 'freelance/craft.html', {'title': 'Создание задачи', 'form':
form})
    # ---Добавить задачи на сайт---

    # Вход в аккаунт
    class LoginUser(DataMixin, LoginView):
        form_class = LoginUserForm
        template_name = 'freelance/vhod.html'
        success_url = reverse_lazy('invate')

        def get_context_data(self, *, object_list=None, **kwargs):
            context = super().get_context_data(**kwargs)
            c_def = self.get_user_context(title="Авторизация")
            return dict(list(context.items()) + list(c_def.items()))

        # Переход после регестрации и входа
        def get_success_url(self):
            return reverse_lazy('home')

        # Выход из аккаунта
        def logout_user(request):
            logout(request)
            return redirect('home')

    # Регистрация на сайт
    class RegisterUser(CreateView, DataMixin):
        form_class = RegisterUserForm
        template_name = 'freelance/register.html'
        success_url = reverse_lazy('register')

        def get_context_data(self, *, object_list=None, **kwargs):
            context = super().get_context_data(**kwargs)
            c_def = self.get_user_context(title="Регистрация")
            return dict(list(context.items()) + list(c_def.items()))
```

Продолжение приложения Б

```
        def get_success_url(self):
            return reverse_lazy('home')

        def form_valid(self, form):
            user = form.save()
            login(self.request, user)
            return redirect('profile')

    # --- Отображение постов в определенной категории ---
    class TaskCatView(ListView):
        model = Task
        template_name = 'freelance/podrubrika.html'
        context_object_name = 'tasks'

        def get_queryset(self):
            return Task.objects.filter(cat__slug=self.kwargs['cat_slug'],
is_published=True)
    # --- Отображение постов в определенной категории ---

    #---- Отображение постов ----#---- Отображение постов ----
    class TaskInfoView(View):

        def get(self, request, post_slug):
            self.profile = Profile.objects.get_or_create(user=request.user)
            post = get_object_or_404(Task, slug=post_slug)
            comment_list = Comment.objects.filter(post_id = post)

            context = {
                    'comment_list': comment_list,
                    'profile': self.profile,
                    'post': post,
                    'title': post.title,
                    'cat_selected': post.cat_id,
                    }
            return render(request, 'freelance/info_task.html', context)
    #---- Отображение постов ----#---- Отображение постов ----

    #--- Изменение постов----
    @login_required
    def edit_task(request, pk):
        task = get_object_or_404(Task, slug=pk)
```

Продолжение приложения Б

```
        if request.method == 'POST':
            task_form = UpdateTaskForm(request.POST, request.FILES, instance=task)
            if  task_form.is_valid():
                task_form.save()
                messages.success(request, 'Your task is updated successfully')
                return HttpResponseRedirect(task.get_absolute_url())
        else:
            task_form = UpdateTaskForm(instance=task)

        return render(request, 'freelance/edit_task.html', {'task_form': task_form, 'title':
"Обновление профиля", })
    #--- Изменение постов----#--- Изменение постов----

    #--- Удаление постов----#--- Удаление постов----
    @login_required
    def delete_task(request,task_id):
        task_delete=Task.objects.get(id=task_id)
        task_delete.delete()
        messages.success(request, 'Your task is delete successfully')
        return
redirect(request.META.get('HTTP_REFERER','redirect_if_referer_not_found'))
    #--- Удаление постов----#--- Удаление постов----
```

Приложение В

Кэширование страниц проекта

```
CACHES = {
   'default': {
      'BACKEND': 'django.core.cache.backends.filebased.FileBasedCache',
      'LOCATION': os.path.join(BASE_DIR, 'taskstart_cache'),
   }
}
```

Импорт метода «cache_page» для кэширования страниц по времени, для обновления новым контентом и быстрой загрузкой из БД.

```
from django.views.decorators.cache import cache_page
```

Для примера поставим 60 секунд и приминим данный метод для страниц содержащие в себе информацию о категориях.

Пример:

«url.py»

```
path('categories/client/design', cache_page(60)(design), name='design'),
path('categories/client/development',cache_page(60)(development),
name='development'),
path('categories/client/seo', cache_page(60)(seo), name='seo'),
path('categories/client/promotion',cache_page(60)(promotion),
name='promotion'),
path('categories/client/audio_video',cache_page(60)(audio_video),
name='audio_video'),
```

РЕЦЕНЗИЯ

на дипломный проект студента 4 курса Казахского национального
исследовательского технического университета им. К.И.Сатпаева
специальности 5В070400 «Вычислительная техника и программное обеспечение»
Дукенбаева Дален Ерболұлы
на тему: «Разработка портала фрилансеров»

Основная идея проекта заключается в разработке системы для пользователей по поиску удаленной работы в Республике Казахстан, целью которой является предоставление взаимовыгодных отношений и общению между фрилансером и заказчиком.

Структура дипломной работы включает в себя: введение, четыре раздела, заключение, список используемых источников литературы и приложений.

Во введении определяется актуальность выбранной темы, цели и задачи исследования, объект и предмет, обосновывается структура дипломной работы.

В первой главе настоящей дипломной работы рассматривается общие представления о проекте, конкретно поставленные цели разработки системы, проанализирован рынок, анализ по предметной области, а также анализ существующих разработок, где были рассмотрены плюсы и минусы их систем.

Во второй главе рассматривается определения требований к системе, а именно требования для начала разработки веб приложения, а именно требования к функционалу, интерфейсу, производительности и требования к безопасности.

В третьей главе рассматривается архитектура системы веб приложения, а также UML модели для проектирования системы.

В четвертой главе обоснован выбор тех или иных инструментов разработок и приведены результаты при разработке системы, а именно разработка пользовательского интерфейса и функционала портала.

В заключении приведены выводы о проделанной работе.

Замечания: имеются технические и стилистические ошибки в тексте дипломного проекта.

Общее заключение и оценка: В целом работа представлена завершенной и может быть оценена на 90 % - «отлично», а при успешной защите Дукенбаева Дален достоин присвоения академической степени бакалавра по специальности 5В070400 «Вычислительная техника и программное обеспечение»

Рецензент:
PhD, директор института
информационных технологий
Некоммерческого акционерного общества
«Алматинский университет энергетики и
связи имени Г. Даукеева» Досжанова А.А.

«____» _____ 20___г.

Қолтаңбаны растаймын
Подпись заверяю
қызметі Алиева З.Ю.
аты-жөні
«16» 05 2022ж.

ОТЗЫВ НАУЧНОГО РУКОВОДИТЕЛЯ

на дипломный проект студента 4 курса Казахского национального исследовательского технического университета им. К.И.Сатпаева специальности 5В070400 «Вычислительная техника и программное обеспечение» Дукенбаев Дален Ерболулы

на тему: «Разработка портала фрилансеров»

В рамках написания данной работы, проведен анализ темы проекта и анализ предметной области, также конкретно определены требования к разработке системы. Разработка портала фрилансеров путь любого разработчика, в любой научной сфере, проявить себя самостоятельно, контролировав тем самым, свое личное время, свой заработок и процесс создания проекта, по заказу клиента.

Высокая практическая ценность работы дополняется лаконичным изложением материала и пояснением проектирования системы, используя как пример диаграммы, а также разъяснением разработки самого портала с помощью выбранных инструментов разработки и хорошо подобранными примерами, что должно вызвать наибольший интерес к работе у специалистов из сферы программного обеспечения.

При выполнении дипломной работы Дукенбаев Д.Е. проявил инициативу и самостоятельность в проведении исследований. Показал себя как вдумчивый, опытный и инициативный специалист, который способен решать различные сложные задачи в ходе проектирования и разработки программного обеспечения. Проявил себя творческим исследователем, способным самостоятельно и на высоком научном уровне выполнять научную работу, обобщать и внедрять полученные результаты. Достаточно углубленно разбирается в современной веб-разработке и в разработке пользовательского обеспечения.

Работа написана логически, последовательно, чётко и ясно. Выполненная работа в полной мере отвечает поставленной цели и является законченным, как в плане анализа по теме проекта, так и по самой разработке системы. Обоснованность и убедительность фактов свидетельствуют о полноте исследований, представленных в научной работе. Оформление работы отвечает принятым стандартам.

Таким образом, данная дипломная работа актуальна, отличается значимой теоретической и практической ценностью, выполнена по всем методическим указаниям написанию и оформлению дипломных проектов. Работа отвечает всем требованиям, предъявляемым к подобным работам, и заслуживает оценки 95 (отлично). . Студент ***Дукенбаев Дален Ерболулы*** заслуживает присвоения академической степени «бакалавр техники и технологии».

Научный руководитель:
PhD, Ассоциированный профессор _____ Н.К. Мукажанов

«17» 05 2022 г.