Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Магнитогорский государственный технический

университет им. Г.И. Носова»

Многопрофильный колледж

Отделение №2 Информационные технологии и транспорт\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ПЦК №10 Информатики и вычислительной техники\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Допустить к защите

Заведующий отделением

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Н.В. Сидорова

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2023 г.

**ДИПЛОМНЫЙ ПРОЕКТ**

**Д.09.02.07.23.ДП.ПЗ**

Обучающейся Аверин Артём Александрович

На тему Умный дом

Состав дипломного проекта:

1. Пояснительная записка на \_\_\_\_ страницах

2. Графическая часть на \_\_\_\_\_\_ листах

Руководитель преподаватель МпК, Зорина И.Г.

Рецензент ИТЦ «Аусферр», инженер-программист 3 категории, Зорин Д.О.

Председатель ПЦК преподаватель МпК, Ремез Т.Б.

(подпись, дата, должность, ученая степень, звание, Ф.И.О.)

|  |  |
| --- | --- |
| Отметка нормоконтролера  \_\_\_\_\_\_\_\_\_\_\_\_\_ /Е.А. Васильева/  «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2023 г. | Выпускница \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись)  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2023 г. |

СОДЕРЖАНИЕ

[**ВВЕДЕНИЕ** 5](#_Toc137558682)

[**1 ТЕОРЕТИКО—ИНФОРМАЦИОННЫЙ АНАЛИЗ ВЕБ—ПРИЛОЖЕНИЯ ПО ПОИСКУ ВАКАНСИЙ** 7](#_Toc137558683)

[**1.1 Анализ предметной области** 7](#_Toc137558684)

[**1.2 Анализ и оценка существующих программных продуктов** 9](#_Toc137558685)

[1.2.1 HeadHunter.ru 9](#_Toc137558686)

[1.2.2 Rabora.ru 10](#_Toc137558687)

[1.2.3 Труд всем 10](#_Toc137558688)

[1.2.4 Авито 11](#_Toc137558689)

[1.2.5 SuperJop 12](#_Toc137558690)

[1.3 Постановка задачи 13](#_Toc137558691)

[**2** **ПРОЕКТИРОВАНИЕ ВЕБ—ПРИЛОЖЕНИЯ ПО ПОИСКУ ВАКАНСИЙ** 15](#_Toc137558692)

[2.1 Моделирование программного продукта 15](#_Toc137558693)

[2.2 Обоснование выбора средств разработки 21](#_Toc137558694)

[2.3 Логическое и физическое проектирование базы данных 23](#_Toc137558695)

[2.4 Разработка базы данных 29](#_Toc137558696)

[2.5 Разработка физической и логической структуры сайта 42](#_Toc137558697)

[2.6 Проектирование меню и пользовательского интерфейса 44](#_Toc137558698)

[2.7 Описание запросов при разработке приложения 45](#_Toc137558699)

[2.8 Проектирование отчетов 54](#_Toc137558700)

[2.9 Администрирование программного продукта 55](#_Toc137558701)

[**3 ОПИСАНИЕ РАБОТЫ ПРОГРАММНОГО ПРОДУКТА** 57](#_Toc137558702)

[3.1 Руководство пользователя 57](#_Toc137558703)

[3.2 Руководство программисту 62](#_Toc137558704)

[3.3 Руководство администратора 64](#_Toc137558705)

[**4 ИНФОРМАЦИОННАЯ БЕЗОПАСНОСТЬ** 66](#_Toc137558706)

[4.1 Общие положение информационной безопасности, уровни информационной безопасности 66](#_Toc137558707)

[Федеральный закон 187 «О безопасности критической информационной инфраструктуры Российской Федерации» 68](#_Toc137558708)

[4.2 Виды возможных угроз веб–приложения по поиску вакансий 69](#_Toc137558709)

[4.3 Информационная безопасность программного продукта 69](#_Toc137558710)

[**5 УПРАВЛЕНИЕ ПРОЕКТОМ** 71](#_Toc137558711)

[5.1 Понятие проекта. Необходимость управления проектом 71](#_Toc137558712)

[5.2 Постановка цели проекта по SMART 72](#_Toc137558713)

[5.3 Анализ заинтересованных сторон проекта 73](#_Toc137558714)

[5.4 Устав проекта 73](#_Toc137558715)

[5.5 Планирование проекта (WBS, диаграмма Ганта) 74](#_Toc137558716)

[5.6 Управление стоимостью 75](#_Toc137558717)

[5.7 Управление рисками проекта 77](#_Toc137558718)

[**ЗАКЛЮЧЕНИЕ** 79](#_Toc137558719)

[**СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ** 80](#_Toc137558720)

[**ПРИЛОЖЕНИЕ А** 82](#_Toc137558721)

# **ВВЕДЕНИЕ**

Трудоустройство является неотъемлемой частью человеческой жизни. Поиск работы зависит от разнообразных факторов, начиная с умений и образования человека, заканчивая требованию работодателей и самой работой.

В последние годы трудоустройство становится тяжелой вещью для людей. Все это происходит из–за нескольких факторов:

* отсутствие опыта работы;
* незнание куда устроится работать;
* отсутствие точного понятия, куда можно применить свои навыки;
* отсутствие работы по образованию.

Основной же проблемой трудоустройства является качественное несоответствие между спросом и предложением рабочей силы, то есть, между структурой имеющихся вакантных мест и профессиональным составом ищущих работу. Хотя увеличение числа заявленных вакансий произошло по всем отраслям экономики, однако, в большей части для замещения свободных мест требуются работники без специальной подготовки по не престижным, низкооплачиваемым профессиями. В городе таких вакансий порядке 86%, причем 655 из них заявлено предприятиями и организациями негосударственного сектора экономики.

Затруднено трудоустройство граждан, имеющих специфические профессии, ранее используемые на предприятиях военно–промышленного комплекса в то же время предприятия города испытывают потребность в специалистах высокой квалификации.

Такая же проблема и с молодыми студентами. Проблема трудоустройства выпускников ВУЗов сегодня чрезвычайно актуальна. Система государственного распределения – ушла в прошлое. Сегодня молодой специалист сталкивается с довольно жесткими условиями рынка, из которых он не всегда выходит победителем. В настоящее время молодежь представляет наиболее многочисленную группу безработных – более 30% от общего числа зарегистрированных безработных. Кроме того, 25—28% от общей совокупности безработной молодежи составляют выпускники учебных заведений.

В современной рыночной экономике проблема занятости является одной из наиболее актуальных. Осуществление глубоких социально–экономических преобразований в стране, стержнем которых является переход к экономике рыночного типа, непрекращающийся процесс совершенствования правового регулирования, обуславливают возникновение целого комплекса новых проблем, в том числе и в сфере занятости населения, требующих своего изучения.

На данный момент поиск работы вышел на новый уровень. Теперь благодаря использованием современных технологий есть возможность подавать свои заявления на работу онлайн. Так же работодатели могут оставлять свои объявления о приёме на работу так же через интернет.

Цель: разработать автоматизированную информационную систему по поиску вакансий.

Для достижения поставленной цели необходимо решить следующие задачи:

* анализ предметной области;
* анализ и оценка существующих программных продуктов;
* постановка задачи;
* моделирование программного продукта;
* логическое и физическое проектирование базы данных
* обоснование выбора средств разработки;
* разработка базы данных;
* разработка физической и логической структуры сайта;
* проектирование меню и пользовательского интерфейса;
* описание запросов при разработке веб-приложения;
* проектирование отчётов;
* администрирование программного продукта;
* руководство пользователя;
* руководство программиста;
* общие положение информационной безопасности, уровни информационной безопасности;
* виды возможных угроз веб-приложения;
* информационная безопасность программного продукта;
* понятие проекта, необходимость управления проектом;
* постановка цели проекта по SMART;
* анализ заинтересованных сторон проекта;
* устав проекта;
* планирование проекта, построение диаграммы WBS и Ганта;
* управление стоимостью;
* управление рисками проекта.

# **1 ТЕОРЕТИКО—ИНФОРМАЦИОННЫЙ АНАЛИЗ ВЕБ—ПРИЛОЖЕНИЯ ПО ПОИСКУ ВАКАНСИЙ**

# **1.1 Анализ предметной области**

Документы неизбежно сталкиваются с обработкой. Под обработкой, понимается сортировка данных, поиск данных по указанной совокупности параметров, добавление, изменение и удаление записей. Сложность обработки данных прямо пропорциональна объему хранимой информации.

Для разрешения проблемы, связанной с обработкой больших объемов данных, необходима разработка информационной системы, обеспечивающей обработку данных из выбранной предметной области. Грамотное проектирование структуры хранения данных и средств их обработки позволяют решить практически любую задачу, связанную с поиском и сортировкой исходного хранилища.

В настоящее время на рынке труда презентовано большее количество всевозможных вакансий, относящихся к различным направлениям коммерческой и иной деятельности. Любая из вакансий характеризуется целостным комплектом показателей, на основании которых определяется возможность принятия на нее того или иного соискателя. При таких размерах хранимой информации реализовать выбор подходящей для соискателя вакансии маршрутом элементарного просмотра каталога вакансий, даже если он и реализован в электронном виде, за разумное время занятие действительно безнадежное. Естественно, что при определенных параметрах отыскивания с учетом принципов компании хранения данных, требуемый итог может быть приобретен довольно скоро (например, при условии упорядочения вакансий по должностям, отбор должностей бухгалтера сможет быть сделан весьма быстро). Впрочем, в случае надобности учета некоторых параметров, обнаружить подходящую вакансию без использования специальных поисковых средств, будет невозможно.

На данный момент рынок труда состоит из двух направлений. В одних сферах деятельности (финансы, юриспруденция, аудит) молодым людям устроиться на работу практически невозможно, так как они еще не обладают глубокими профессиональными знаниями и значительным опытом. Зато в других областях (торговля, маркетинг, шоу–бизнес) их молодость, сила и энергичность, а главное – еще не очень большие материальные запросы – воспринимаются работодателями на «ура». Часто работодатель отдает предпочтение молодым, и в том числе несовершеннолетним, исключительно в целях экономии: выполняют работу наравне с взрослыми, а платить можно в два, а то и в три раза меньше.

В категории молодежь можно выделить, по меньшей мере, 4 подгруппы (14–16–летние, 16–18–летние, 18–23–летние, старше 23 лет), каждая из которых имеет свои специфические черты, которые необходимо учитывать при трудоустройстве молодого человека в соответствии с российским законодательством. Для двух групп молодежи (18–23–летние и старше 23 лет) целесообразны мероприятия, направленные на построение эффективных самостоятельных взаимоотношений с участниками рынка труда, стимулирование развития творческой активности молодежи и ее ответственности за свое будущее.

Основополагающим инструментом государственного регулирования занятости населения выступает нормативно–правовая база. Законодательство о занятости не только регулирует трудовые отношения и обеспечивает правовую защищенность работника, но и оказывает непосредственное влияние на эффективность функционирования рынка труда: динамику занятости и безработицы, перераспределение рабочей силы и рабочих мест, а также благосостояние отдельных социальных групп.

Существующее законодательство не предоставляет на сегодняшний день дополнительных гарантий в области трудоустройства и занятости населения для молодежи, особо нуждающейся в социальной защите и испытывающих трудности в поиске работы, – например, обязательное квотирование рабочих мест для данных категорий граждан.

Вопрос занятости населения  в настоящие время является  одним из самых обсуждаемых, государством разрабатываются различные программы социальных реформ, в которых делается упор на совершенствование рыночных механизмов регулирования занятости:

* программы по стимулированию роста занятости  и увеличению числа рабочих  мест;
* программы, направленные на подготовку и  переподготовку рабочей силы;
* программы содействия найму рабочей силы.

Из всего вышесказанного можно сделать вывод о том, что проблема занятости населения и безработицы на сегодняшний день является весьма актуальным вопросом.

Соответственно, реализация автоматизированной информационной системы, обеспечивающей поиск вакансий по трудоустройству по произвольной совокупности их данных за рациональное время, представляется непременным условием наличия бюро как такового.

Есть два вида пользователей, которые пользуются веб-приложениями по поиску вакансий. Соискатель – это человек, который занят поиском место труда, в зависимости от его образования, навыков и желаний. Соискатель ищет работу у работодателей. Цель работодателя найти работника для своего бизнеса.

Основная функция веб–приложений для поиска работы состоит в анализе и прогнозировании спроса предложений на работу в разных сферах жизнедеятельности человека. Так же возможность оказать помощь гражданам в выборе подходящей работы, в зависимости от их обучения, желания или опыта. Для работодателей это работает в направлении возможности в подборе необходимых работников, в зависимости от требований работодателей.

Данные веб–приложения могут не только принять на работу опытных специалистов, но и начинающих желающих, давая им возможность пройти обучение или подготовку той или иной специальности, в зависимости от разнообразных факторов.

# **1.2 Анализ и оценка существующих программных продуктов**

Уже существуют сайты для поиска работы онлайн. И некоторые их них действительно хорошо и популярны, и пользуются большим спросом у аудитории. Но не один сайт не может быть идеален, в каждом из них есть свои недочёты и ошибки. В данном случае будут рассмотрены несколько сайтов, которыми пользуются исключительно в России.

# **1.2.1 HeadHunter.ru**

HeadHunter –лидер по популярности среди российских веб–сайтов по поиску работы. Достоинством сайта является, пожалуй, самая большая среди конкурентов база резюме и вакансий. Кроме того, сервис отличается интуитивно понятной функциональностью, а именно: удобной системой поиска, системой отслеживания откликов и просмотров объявления. Так же стоит отметить, что сайт приятен на внешний вид (рис. 1).

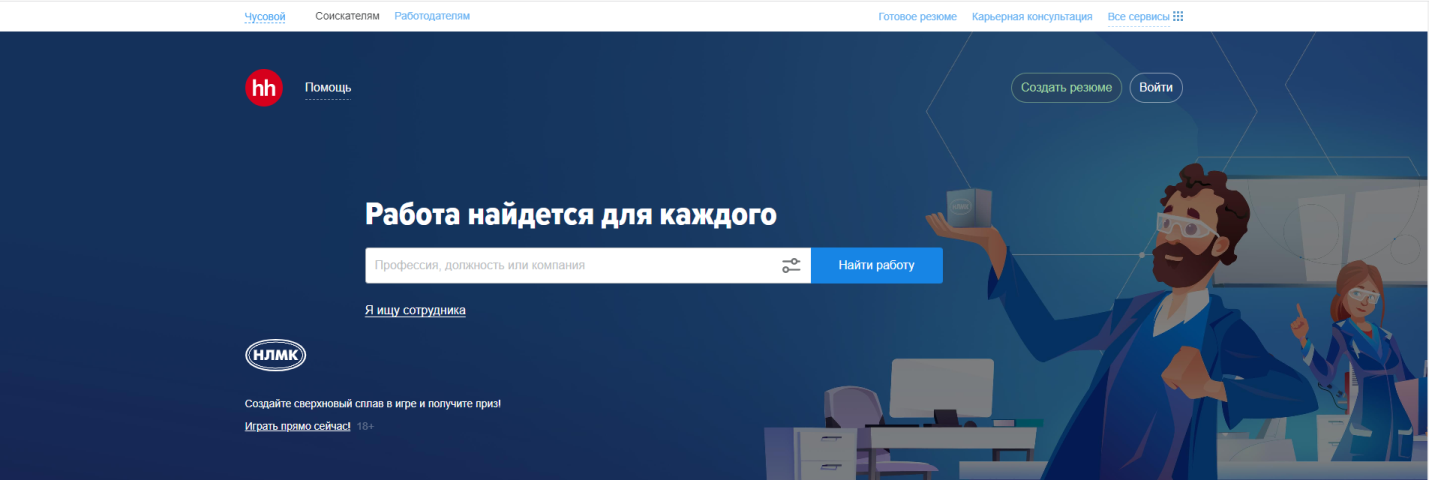


Рисунок 1 — Сайт HeadHunter.ru

## 1.2.2 Rabora.ru

Данный сайт является вторым по популярности в России. Он отличается удобным разделением вакансий по категориям, например, «работа в строительстве», «работа в медицине» и т.д. Кроме того, есть раздел с вакансиями, подходящими для людей с инвалидностью. Rabota.ru схожа с HeadHunter наличием строго регламента по добавлению вакансий и резюме, наличием формы для заполнения полей. Кроме того, на указанных сайтах компании проходят верификацию, так что риск столкнуться с мошенниками минимизируется.

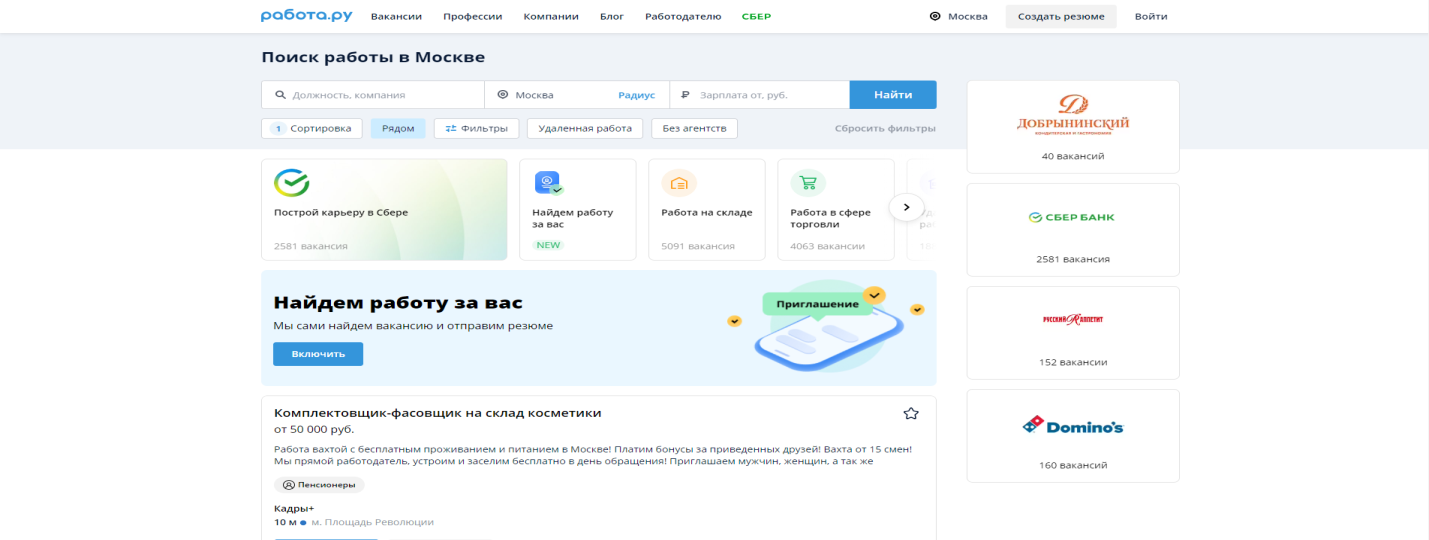


Рисунок 2 — Сайт Rabora.ru

# **1.2.3 Труд всем**

Ресурс создан Федеральной службой по труду и занятости. При создании аккаунта проходит синхронизация с Госуслугами. Информация о предприятиях, единая база данных и комфортная навигация по сайту – главные плюсы сервиса, по мнению пользователей. Касательно содержания вакансий, несмотря на «статус» создателей, ресурс не смог выстроить системы без мошенников и сомнительных предложений, которые нередко встречаются на популярных площадках.

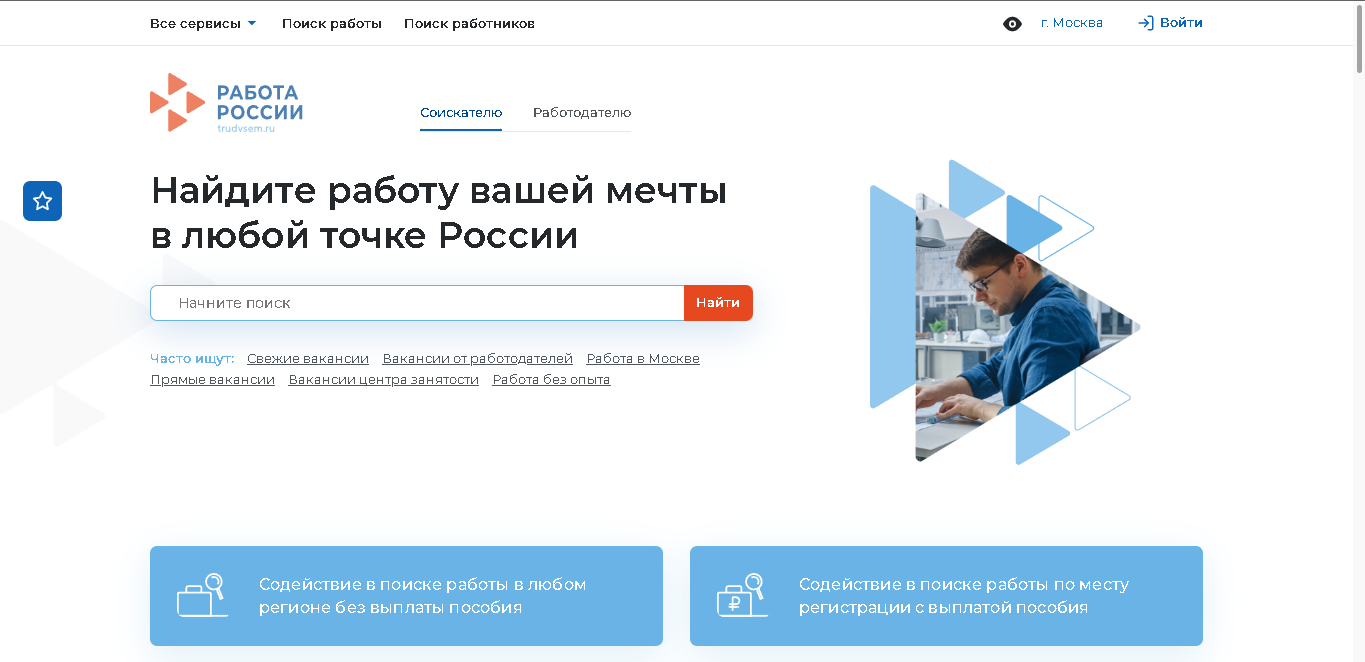


Рисунок 3 — Сайт Труд всем

## 1.2.4 Авито

Раздел с вакансиями на Авито отличается легкой системой добавления объявлений. Отсутствуют регламент и форма для добавления информации, объявления проходят модерацию, но компании работодатели никак не проверяются, поэтому легкость в добавлении объявлений компенсируется большим риском столкновения с мошенниками. Avito отличается также упрощенной системой фильтрации, на сайте можно отсортировать вакансии лишь по основным признакам, таким как график работы и зарплата.

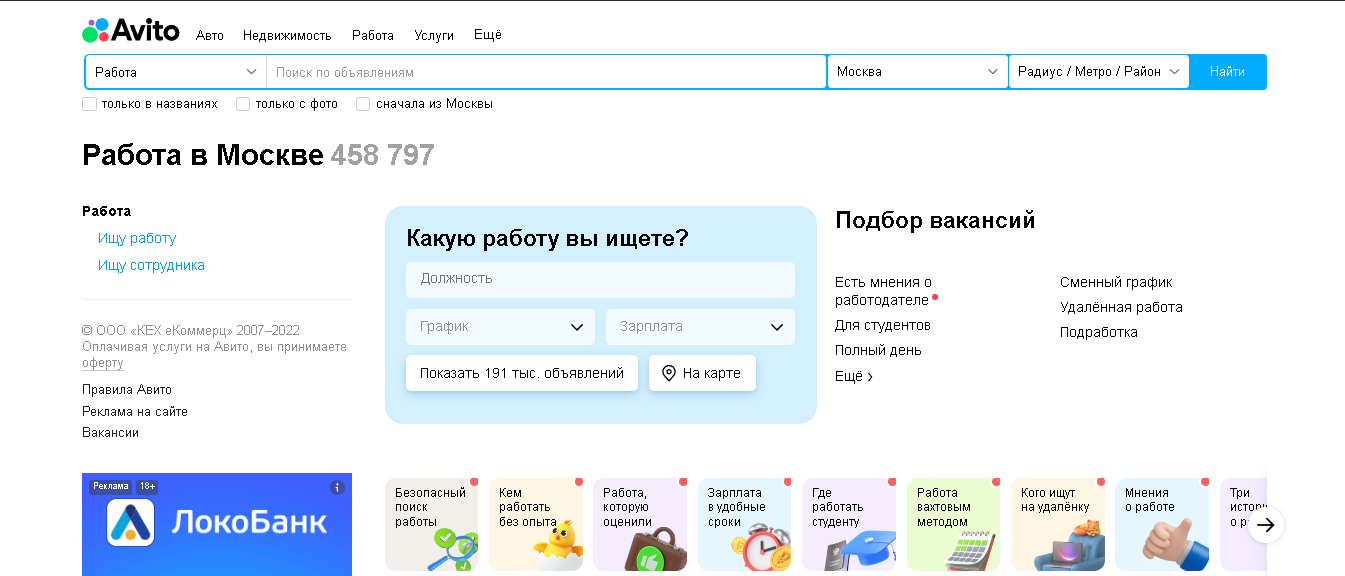


Рисунок 4 — Сайт Авито

## 1.2.5 SuperJop

Вначале данный сайт был похож на Авито своим статусом «доска объявлений», а теперь сайт занимает одну пятую доли сервисов поиска работы в стране. Они первые добавили бесплатные видеокурсы, которые помогут соискателям добиться трудоустройства.

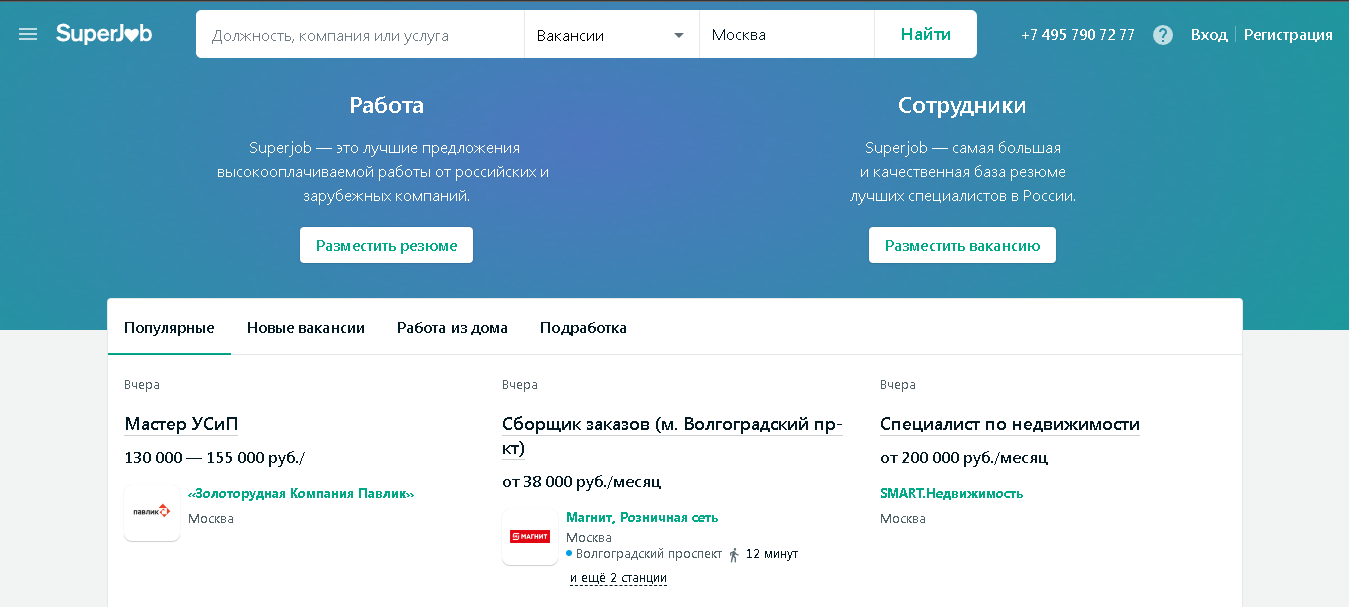


Рисунок 5 — Сайт SuperJop

После ознакомления с вышеперечисленными существующими программными продуктами были затронуты характеристики, плюсы и минусы веб приложений, показанные в сравнительной таблице 1.

Таблица 1 – Сравнение существующих программных продуктов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Факторы | Hh.ru | Rabota.ru | Труд всем | Авито | SuperJob |
| Привлекательный дизайн | + | — | — | — | — |
| Большое количество заявок | + | — | — | + | — |
| Оптимизация под мобильное устройство | + | — | — | + | + |
| Обширный функционал | + | — | — | + | — |

Продолжение таблицы 1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Расширенный поиск | + | + | — | + | — |
| Обширная информация о вакансии | + | + | — | + | — |
| Спам | + | + | + | + | + |
| Дублирование объявлений | — | + | + | — | + |
| Реклама | + | + | + | + | + |

Вследствие, после проведения сравнительной таблицы выяснилось, что основной проблемой данных сайтов является малое количество объявлений, тем самым предлагая малый выбор для соискателей. Также стоит отметить, что все анализируемые сайты содержат в себе спам–рекламу, что мешает только его работе. При это не все сайты могут похвастаться притягательным дизайном для пользователей.

Самой большой проблемой данных сайтов является частое дублирование объявлений на сайте от одного работодателя, что только мешает соискателям и засоряет веб–приложение.

## 1.3 Постановка задачи

Требуется разработать веб–приложение по поиску вакансий, на котором соискатели смогу найти подходящую им вакансию, а работодатели выложить свои объявления на работу. Функционал веб-приложения должен различаться в зависимости от роли зарегистрированного пользователя.

В веб-приложении работодатель может создавать свои объявления. После, эти объявления могут редактироваться, скрываться с сайта, удаляться, и просматриваться сообщения к данным объявлениям.

Соискатель может редактировать своё резюме, которое он заполняет при регистрации в веб-приложении, а так же может добавлять и изменять фотографию профиля. Соискатели могут оставлять лайки к понравившимся объявлениям, а так же отправлять отклики с сообщениями.

Функционал администратора состоит из просмотра всех объявлений, с возможность их удалять и скрывать на сайте. Так же, администратор может заблокировать или разблокировать соискателей или работодателей. Главной частью функционала администратора является возможность сравнения показателей сайта за определённые периоды.

Цель веб–приложения заключается в создании удобного, простого и безопасного веб–приложения по поиску вакансий, которое сможет предоставить обширный выбор вакансий, разнообразный функционал, обеспечивая способность выставлять заявки и возможность провести анализ области вакансий с административной стороны пользования.

Задачи, которые необходимо выполнить:

* разработать регистрацию пользователей;
* разработать возможность добавлять устройства;
* осуществить возможность редактировать и удалять устройства;
* разработать возможность скачать резюме соискателей после принятия их отклика в pdf формате;
* создать расширенный поиск для объявлений;
* разработать возможность соискателям редактировать своё резюме и свою фотографию профиля;
* разработать возможность блокировать или разблокировать пользователей веб-приложения со стороны администратора;
* осуществить возможность удалить или скрыть объявления со стороны администратора;
* разработать логистику сайта со стороны администратора.

# **2 ПРОЕКТИРОВАНИЕ ВЕБ—ПРИЛОЖЕНИЯ ПО ПОИСКУ ВАКАНСИЙ**

## 2.1 Моделирование программного продукта

UML – это универсальный язык визуального моделирования систем. Он имеет широкое применение благодаря свойственной ему расширяемости. UML объединил лучшие современные технические приемы моделирования и разработки программного обеспечения. По сути, язык UML был задуман так, чтобы его можно было реализовать посредством его же инструментальных средств. Фактически это признание того, что большие современные программные системы, как правило, нуждаются в инструментальной поддержке. UML–диаграммы легко воспринимаются.

UML способен предоставить поддержку визуального моделирования для разработки программного продукта.

Неизменная цель UML – способствовать объединению всего лучшего в опыте разработки программного обеспечения последнего десятилетия. Для этого UML унифицируют опыт предшествующих языков визуального моделирования и процессов разработки программного обеспечения наиболее оптимальным образом.

UML прилагает усилия в унификации нескольких разных областей:

* жизненный цикл разработки: UML предоставляет визуальный синтаксис для моделирования на протяжении всего жизненного цикла разработки программного обеспечения – от постановки требований до реализации;
* области приложений: UML используется для моделирования всех аспектов – от аппаратных встроенных систем реального времени до систем поддержки принятия решений;
* языки реализации и платформы: UML является независимым от языков и платформ;
* процессы разработки: UML может поддерживать множество других процессов разработки программного обеспечения;
* собственные внутренние концепции: UML поистине стойко стремится сохранить последовательность и постоянство применения не­ большого набора своих внутренних концепций. До сих пор это не всегда удавалось, но в этом направлении наблюдается заметный прогресс по сравнению с предыдущими попытками.

Диаграмма прецедентов.

Диаграмма прецедентов представляет собой описание последовательности действий, включая альтернативные и ошибочные последовательности, которые система, подсистема или класс могут осуществлять, взаимодействия с внешними актерами. Диаграмма прецедентов описывает поведение, демонстрируемое системой с целью получения значимого результата для одного или более актеров. Диаграмма прецедентов предоставлена на рисунке 6.

Прецедент – это что–то, что должна делать система по желанию актера. Это «вариант использования» системы конкретным актером:

* прецеденты всегда инициируются актером;
* прецеденты всегда описываются с точки зрения актеров

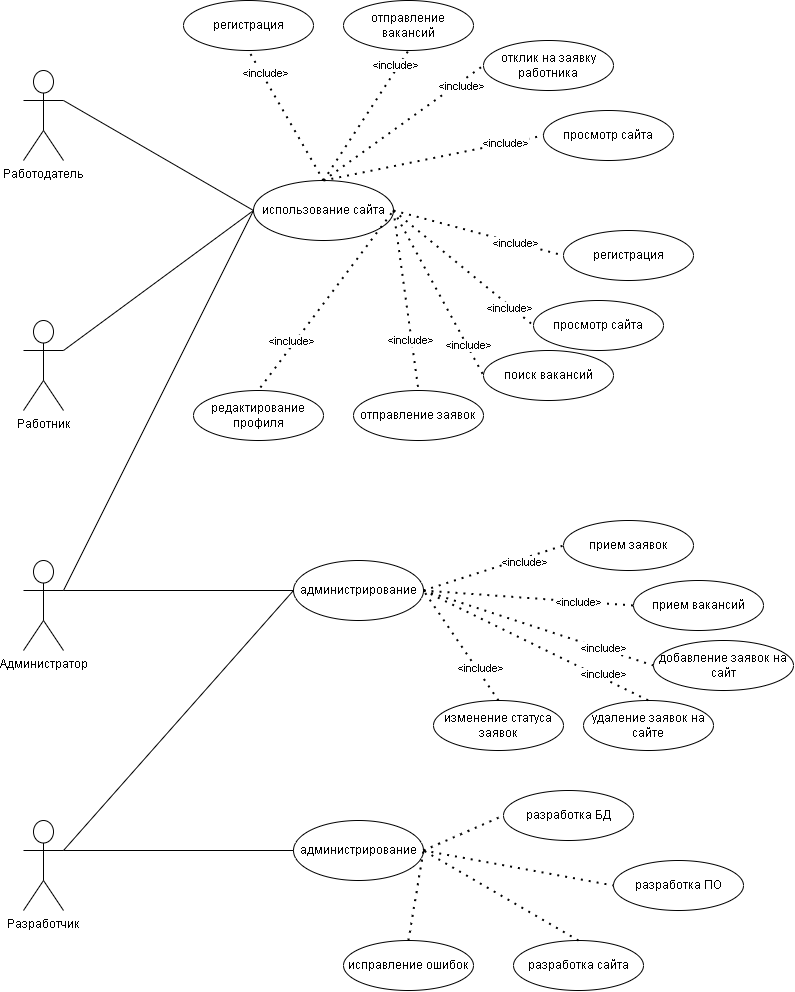


Рисунок 6 — Диаграмма прецедентов

Диаграмма компонентов – это, фактически, список артефактов, из которых состоит моделируемая система, с указанием некоторых отношений между артефактами. Наиболее существенным типом артефактов программных систем являются программы. Таким образом, на диаграмме компонентов основной тип сущностей – это компоненты (как исполнимые модули, так и другие артефакты), а также интерфейсы (чтобы указывать взаимосвязь между компонентами) и объекты (входящие в состав компонентов). На диаграмме компонентов применяются следующие отношения: Диаграмма компонентов показана в рисунке 7

* реализация между компонентами и интерфейсами (компонент реализует интерфейс);
* зависимости между компонентами и интерфейсами (компонент использует интерфейс);
* зависимости между объектами и компонентами (объект входит в компонент).

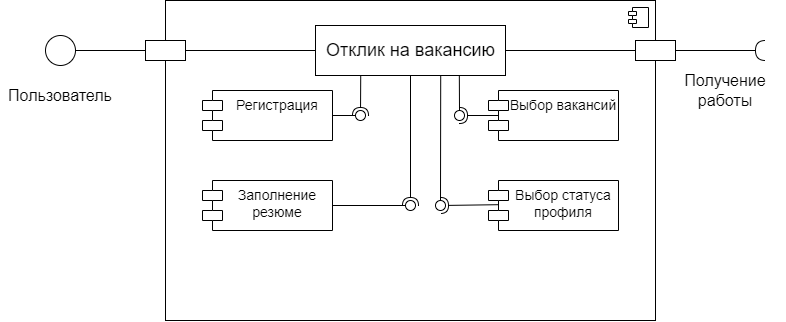


Рисунок 7 — Диаграмма компонентов

Диаграмма деятельности позволяет моделировать процесс как деятельность, которая состоит из коллекций соединенных ребрами узлов.

Деятельность может быть добавлена к любому элементу модели с целью моделирования его поведения. Элемент обеспечивает контекст для деятельности, и деятельность может использовать возможности своего контекста. Деятельности обычно добавляются к:

* прецедентам;
* классам;
* интерфейсам;
* компонентам;
* кооперациям;
* операциям.

Диаграммы деятельности также могут использоваться для моделирования бизнес–процессов и рабочих потоков. Диаграмма деятельности изображена на рисунке 8.

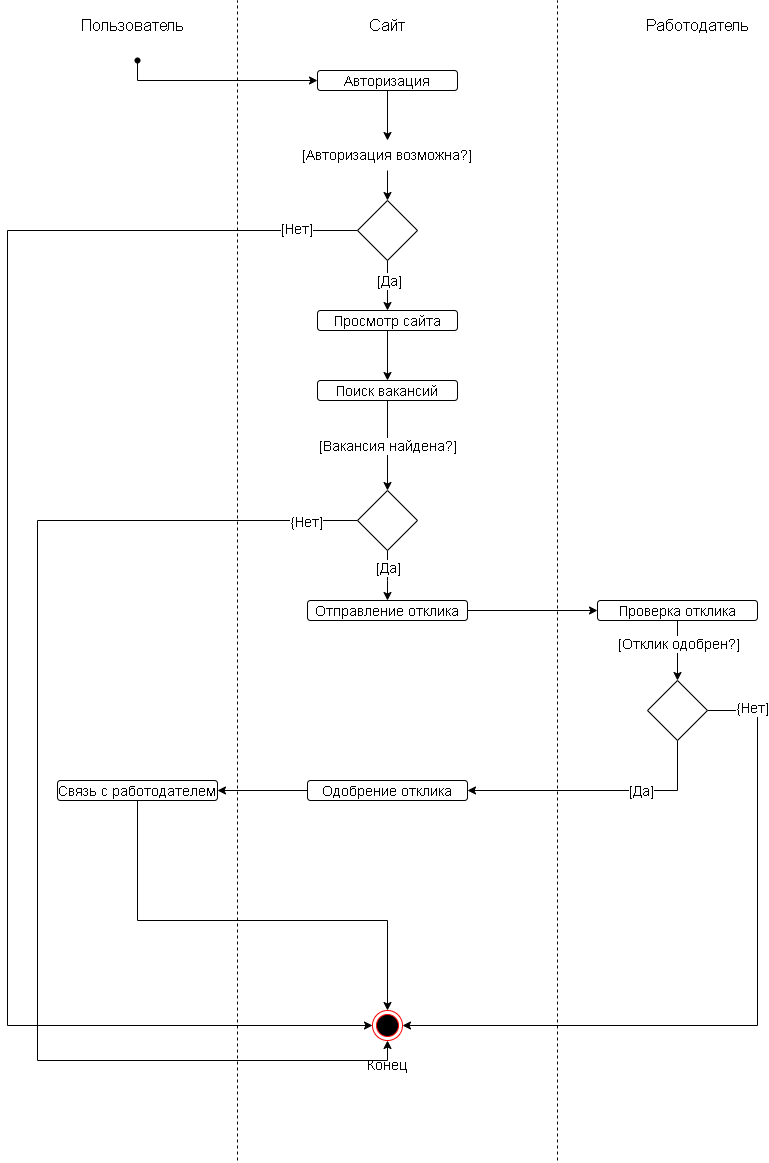


Рисунок 8 — Диаграмма деятельности

Диаграмма последовательностей.

Диаграммы последовательностей представляют взаимодействия между линиями жизни как упорядоченную последовательность событий. Это самая богатая и гибкая форма диаграммы взаимодействий. Диаграмма последовательностей представлена на рисунке 9.

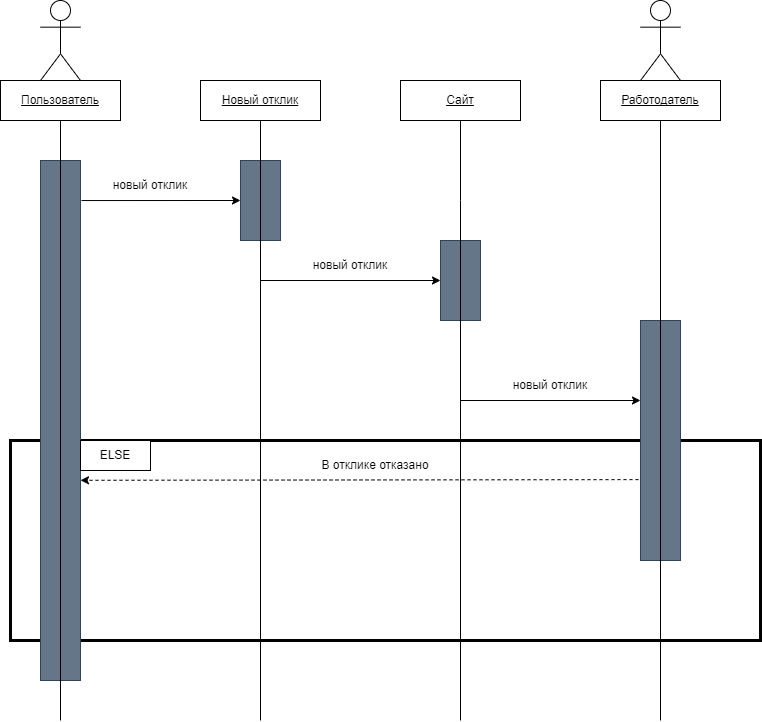


Рисунок 9 — Диаграмма последовательности

Диаграмма состояний

Диаграмма состояний представляет собой переход объекта, который постоянно переходит из одного состояния в другое. Диаграмма состояний содержит только один конечный автомат для единственного реактивного объекта, как показано на рисунке 10.

У каждого конечного автомата должно быть начальное состояние, обозначающее первое состояние последовательности. Если смена состояний не бесконечна, должно присутствовать и конечное состояние, которое завершает последовательность переходов. Обычно переход от начального псевдосостояния к первому «настоящему» состоянию происходит автоматически.

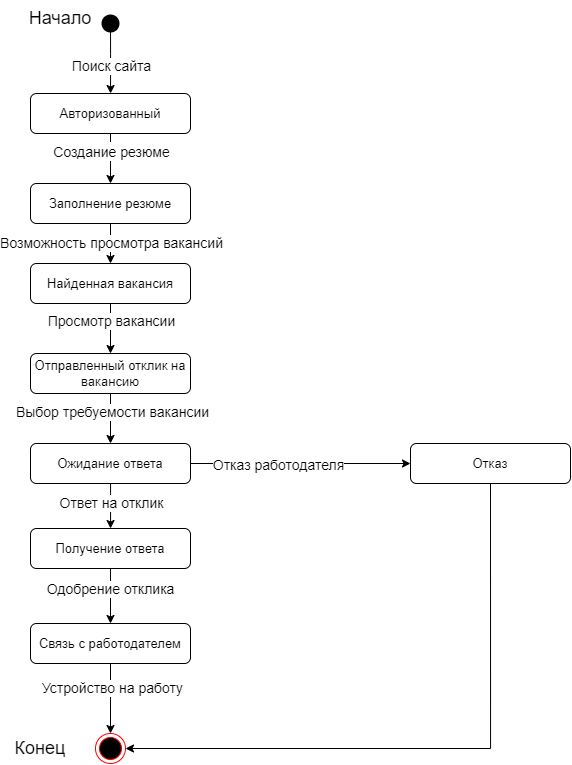


Рисунок 10 — Диаграмма состояния

## 2.2 Обоснование выбора средств разработки

В процессе создания веб–приложения были выбранные различные средства для разработки.

Выбранные средства для разработки:

* MySQL;
* Laravel.

MySQL – это свободно распространяемая система управления базами данных. MySQL имеет клиент–серверную архитектуру: к серверу MySQL могут обращаться различные клиентские приложения, в том числе с удаленных компьютеров.

MySQL имеет ряд плюсов, за счет которых она была выбрана для разработки:

* MySQL представляет из себя открытый код. Любой желающий может бесплатно скачать программу и при необходимости доработать ее;
* MySQL – кроссплатформенная система. Ее можно использовать практически во всех современных операционных системах;
* MySQL имеет множество программных интерфейсов, благодаря которым к базе данных MySQL могут подключаться приложения, созданные с помощью различных языков, таких как: C/C++, PHP Python и так далее;
* MySQL имеет отличные технические характеристики: многопоточность, многопользовательский доступ, быстродействие, масштабируемость
* MySQL имеет развитую систему обеспечения безопасности и разграничения доступа на основе системы привилегий

MySQL представляет из себя прекрасное средство для разработки веб–приложения, которая содержит сбалансированный набор функций, возможностей и преимущества. MySQL идеально подходит для быстродействия при взаимодействии данных, он понятен и прост в управлении системе управления базами данных.

Bootstrap – это бесплатный CSS–фреймворк с открытым исходным кодом, предназначенный для быстрой верстки адаптивных интерфейсов и веб–приложений с ориентацией на мобильные устройства.

На данный момент Bootstrap является одним из самых распространённых фреймворков для проектирования веб–приложений. Он пользуется популярностью не только у разработчиков сайтов, но и у заказчиков.

Bootstrap содержит в себе плюсы, которые обеспечивают быструю и качественную разработку веб–приложения, такие как:

* уменьшение количества времени, затрачиваемого на разработку. Используя Bootstrap,его библиотеки и документацию, позволяет экономить время, ведь он даёт готовые решения и шаблоны;
* Bootstrap позволяет создавать адаптивные веб–приложения, и дизайн будет корректно отображаться на экранах устройств разных размеров вне зависимости от их диагонали и размеров;
* веб–приложения, сделанные на Bootstrap, будут одинаково отображаться во всех современных браузерах;
* простой и понятный код позволяет ознакомиться в фреймворков быстро и легко, так как содержит качественный и структурированный код;
* элементы Bootstrap смотрятся гармонично между собой и позволяют создавать веб–приложение в едином стиле.

Исходя из вышеперечисленных плюсов, Bootstrap показал себя как качественный фреймворк, который будет прост в изучении и применении для разработки веб–приложения.

Laravel является одним из самых популярных в мире PHP–фреймворков, предназначенный для разработки веб–приложений. Имя при себе ряд плюсов, стоит выделить некоторые их низ:

* Laravel – полнофункциональный фреймворк. Он содержит все программные подсистемы необходимые для разработки веб–приложения. После его установки нет необходимости доустанавливать ещё какие–либо программы или приложения;
* у Laravel низкий порог вхождения. Для программирования веб–приложений хватает базовых знаний PHP и основ веб–разработки. Он не требует сложного конфигурирования и готов к работе сразу после установки. Отдельные модули, составляющие код, не требуется явно связывать друг с другом, достаточно «разложить» их по нужным папкам и веб–приложение будет прекрасно работать;
* Laravel содержит множество дополнительных библиотек, расширяющих функциональность фреймворка, и программ, помогающих в работе.

Таким образом, фреймворк Laravel представляет из себя качественный продукт, которые облегчит разработку веб–приложения, поможет сжать код и обеспечить разработку всего требуемого функционала для веб–приложения.

## 2.3 Логическое и физическое проектирование базы данных

Логическим проектированием баз данных формально можно считать процесс создания модели мира самой базы данных, без той системы, которая будет позволять ей работать, и прочих физических деталей. Точность и полнота играют в этом процессе ключевую роль. Одним из главных преимуществ этого этапа является то, что всегда можно взять черновой проект, отложить его в сторону и начать все заново или просто внести желаемые поправки. Гораздо легче менять те или иные детали на этапе проектирования, чем иметь дело с проблемами уже реализованной производственной базы данных, которая плохо спроектирована.

Этап логического проектирования иногда делят на концептуальную и логическую часть, но это отличие чисто номинально. Концептуальное проектирование базы данных обычно предшествует этапу логического проектирования и подразумевает моделирование информации без использования какой–либо базовой модели данных. Что касается этапа логического проектирования, то он уже предполагает явное применение конкретной модели данных, например, реляционной, и фокусирование внимания на логических отношениях, которые были определены на этапе концептуального проектирования. В частности, **логическое проектирование** заключается в **концептуальном моделировании базы данных** и гарантии того, что данные в таблицах проходят проверку на целостность и не являются избыточными. Для удовлетворения этих требований реализуются принципы нормализации данных, о которых будет более подробно рассказываться чуть позже.

Методика, которую чаще всего применяют для логического представления и анализа компонентов бизнес–системы и моделирования подходящего решения после завершения анализа требований, называется ER–моделированием. ER–модели легко конструировать, а, благодаря акценту на графическом представлении, еще и очень легко понимать. Польза ER–моделирования заключается в проектировании, а не в реализации баз данных. Образовывать основу для высокоуровневого языка манипулирования данными вроде SQL ER–моделирование не позволяет, поэтому та модель, которую проектировщики создают с помощью ER–моделирования, для реализации преобразуется в реляционную модель. То есть за счет преобразования абстрактной структуры с сущностями и отношениями в схему реляционной базы данных реляционная модель помогает преобразовывать ER–модель в реляционную систему управления базами данных.

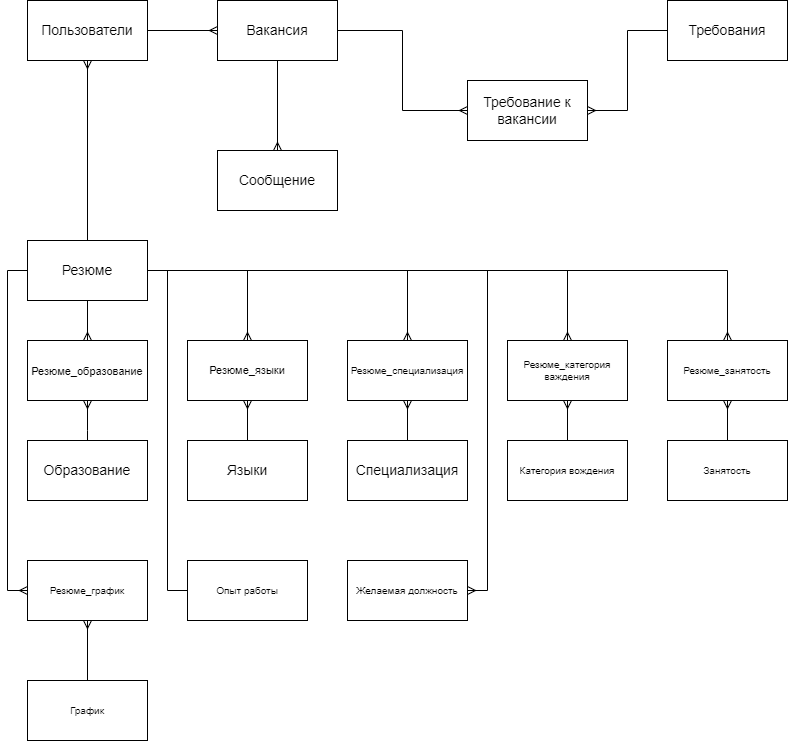


Рисунок 11 — ER—модель базы данных

Можно выделить три основные виды логических моделей:

* иерархическую модель;
* сетевую модель;
* реляционную модель.

Для логической модели данных характерно то, что выполняя все основные требования, предъявляемые системой управления базой данных, не поддерживается ориентация на конкретную систему управления базой данных, что реализуется в физической модели данных.

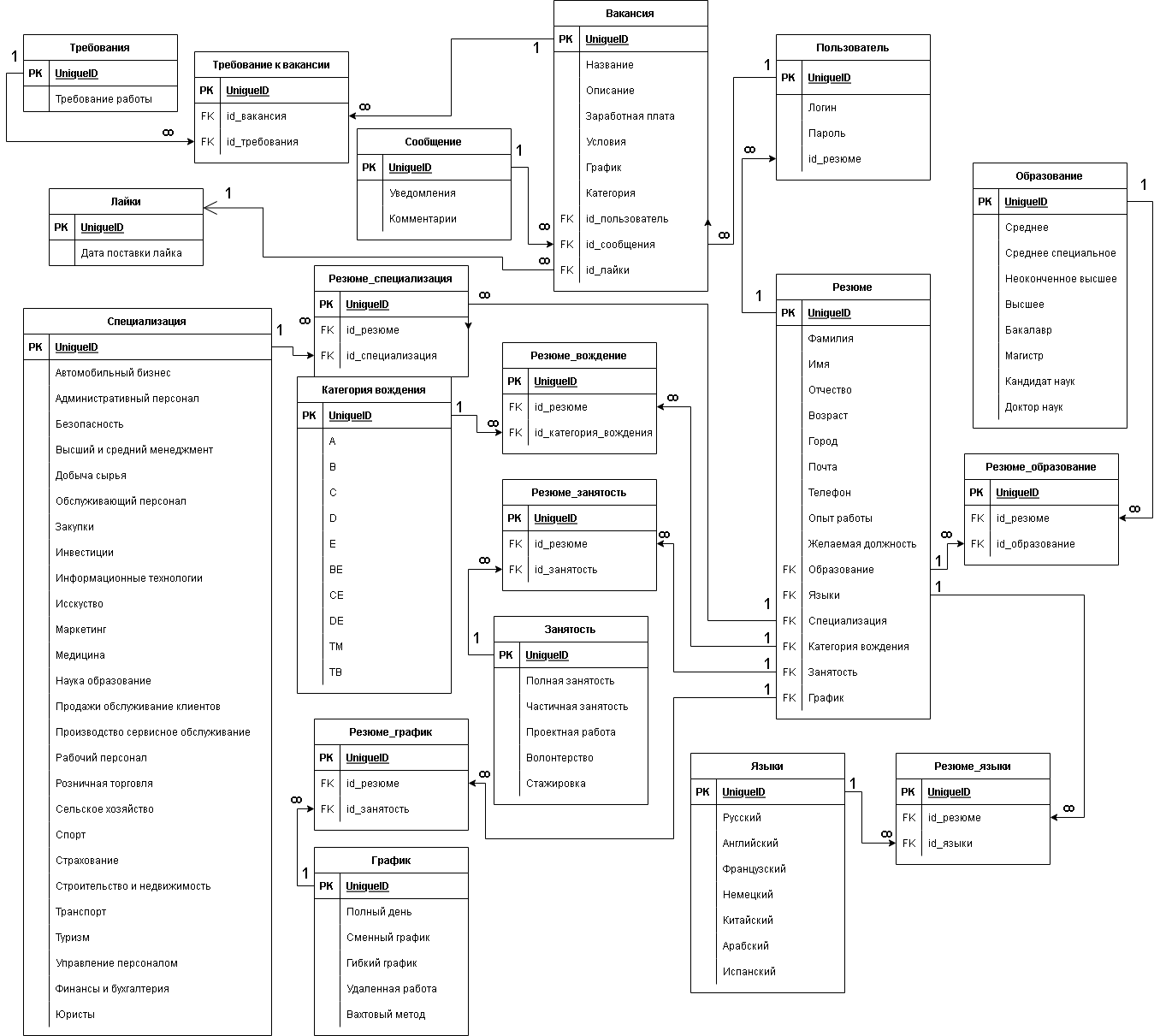


Рисунок 12 — Логическая модель базы данных

Физическая модель данных описывает данные средствами конкретной системы управления базой данных. Ограничения, имеющиеся в логической модели данных, реализуются различными средствами системы управления базами данных, например, при помощи индексов, декларативных ограничений целостности, триггеров, хранимых процедур. При этом опять–таки решения, принятые на уровне логического моделирования определяют некоторые границы, в пределах которых можно развивать физическую модель данных. Точно также, в пределах этих границ можно принимать различные решения. Например, отношения, содержащиеся в логической модели данных, должны быть преобразованы в таблицы, но для каждой таблицы можно дополнительно объявить различные индексы, повышающие скорость обращения к данным. Многое тут зависит от конкретной системы управления базы данных.

Если физическая модель данных реализована средствами реляционной системой управления базами данных, то отношения, разработанные на стадии формирования логической модели данных, преобразуются в таблицы, атрибуты становятся столбцами таблиц, для ключевых атрибутов создаются уникальные индексы, домены преображаются в типы данных, принятые в конкретной системе управления базой данных.

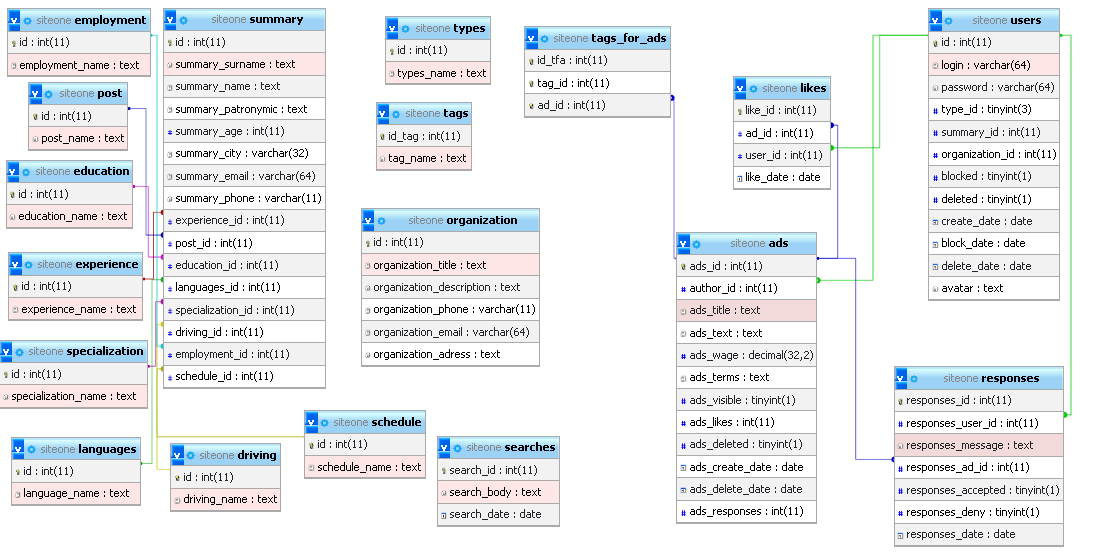


Рисунок 13 – Физическая модель базы данных

Таблица «ads» содержит информацию о вакансиях.

Структура таблицы «ads» (объявления):

* «id» — идентификатор;
* «users\_id» — внешний ключ к таблице «users»;
* «title» — наименование объявления;
* «text» — описание заявки;
* «wage» — заработная плата;
* «terms» — условия заявки;
* «visible» — информация о отображении объявления;
* «likes» — внешний ключ к таблице «likes»;
* «create\_date» — дата создания объявления;
* «responses» — внешний ключ к таблице «responses».

Таблица «driving» содержит информацию об опыте вождения.

Структура таблицы «driving» (опыт вождения):

* «id» — идентификатор;
* «name» — тип опыта вождения.

Таблица «education» содержит информацию об имеющемся образовании.

Структура таблицы «education» (образование):

* «id» — идентификатор;
* «name» — вид образования.

Таблица «employment» содержит информацию о виде занятости.

Структура таблицы «employment» (занятость):

* «id» — идентификатор;
* «name» — вид занятости.

Таблица «experience» содержит информацию об опыте работы соискателя.

Структура таблицы «experience» (опыт):

* «id» — идентификатор;
* «name» — вид опыта.

Таблица «languages» содержит информацию о знании языка соискателя.

Структура таблицы «languages» (языки):

* «id» — идентификатор;
* «name» — наименование языка.

Таблица «languages» содержит информацию о лайках поставленных соискателем на объявление.

Структура таблицы «languages» (языки):

* «id» — идентификатор;
* «ads\_id» — внешний ключ к таблице «ads»;
* «users\_id» — внешний ключ к таблице «users»;
* «date» — дата поставки лайка.

Таблица «organization» содержит информацию об организации.

Структура таблицы «organization» (организация):

* «id» — идентификатор;
* «title» — наименование организации;
* «description» — описание организации;
* «phone» — номер телефона организации;
* «email» — электронная почта организации;
* «adress» — адрес организации.

Таблица «post» содержит информацию о желаемой должности соискателя.

Структура таблицы «post» (должность):

* «id» — идентификатор;
* «name» — наименование желаемой должности.

Таблица «responses» содержит информацию о сообщении от соискателя к заявкам.

Структура таблицы «responses» (сообщения):

* «id» — идентификатор;
* «user\_id» — внешний ключ на таблицу user;
* «message» — сообщение к заявке;
* «ads\_id» — внешний ключ на таблицу заявки.

Таблица «search» содержит информацию о желаемом графике работы для соискателя.

Структура таблицы «schedule» (график):

* «id» — идентификатор;
* «name» — наименование графика.

Таблица «specialization» содержит информацию о специализации соискателя.

Структура таблицы «specialization» (специализация):

* «id» — идентификатор;
* «name» — наименование специализации.

Таблица «summary» представляет из себя резюме соискателя.

Структура таблицы «summary» (резюме):

* «id» — идентификатор;
* «surname» — наименование специализации.
* «name» — наименование специализации.
* «patronymic» — наименование специализации.
* «age» — наименование специализации.
* «city» — наименование специализации.
* «email» — наименование специализации.
* «phone» — наименование специализации.
* «experience\_id» — внешний ключ на таблицу заявки;
* «post\_id» — внешний ключ на таблицу заявки;
* «education\_id» — внешний ключ на таблицу образование;
* «languages\_id» — внешний ключ на таблицу языки;
* «specialization\_id» — внешний ключ на таблицу специализация;
* «driving\_id» — внешний ключ на таблицу вождение;
* «employment\_id» — внешний ключ на таблицу занятость;
* «schedule\_id» — внешний ключ на таблицу график.

Таблица «tags» содержит информацию о тегах.

Структура таблицы «tags» (теги):

* «id» — идентификатор;
* «name» — наименование тега.

Таблица «tags\_for\_ads» содержит информацию о тегах, добавленных к объявлениям.

Структура таблицы «tags» (теги):

* «id» — идентификатор;
* «tag\_id» — внешний ключ на таблицу «tags»;
* «ads\_id» — внешний ключ на таблицу «ads»;

Таблица «types» содержит информацию о типе зарегистрированного пользовать, является он соискателем, работодателем или администратором.

Структура таблицы «types» (тип):

* «id» — идентификатор;
* «name» — наименование типа.

Таблица «users» содержит информацию о зарегистрированном пользователе.

Структура таблицы «users» (тип):

* «id» — идентификатор;
* «login» — логин пользователя;
* «password» — пароль пользователя;
* «type\_id» — внешний ключ на таблицу тип;
* «summaty\_id» — внешний ключ на таблицу резюме;
* «ordanization\_id» — внешний ключ на таблицу организация;
* «blocked» — информации о блокировке пользователя;
* «delete» — информации об удаление пользователя;
* «create\_date» — дата добавление пользователя;
* «avatar» – фотография пользователя.

## 2.4 Разработка базы данных

Создание таблицы Ads(Объявления), характеристики полей представлены в таблице 2.

CREATE TABLE `ads` (

`ads\_id` int(11) NOT NULL AUTO\_INCREMENT,

`author\_id` int(11) NOT NULL,

`ads\_title` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

`ads\_text` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

`ads\_wage` decimal(32,2) NOT NULL,

`ads\_terms` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

`ads\_visible` tinyint(1) NOT NULL,

`ads\_likes` int(11) NOT NULL,

`ads\_deleted` tinyint(1) NOT NULL,

`ads\_create\_date` date NOT NULL,

`ads\_delete\_date` date NOT NULL,

`ads\_responses` int(11) NOT NULL,

PRIMARY KEY (`ads\_id`),

UNIQUE KEY `ads\_id` (`ads\_id`),

KEY `author\_id` (`author\_id`),

KEY `ads\_likes` (`ads\_likes`),

KEY `ads\_responses` (`ads\_responses`),

CONSTRAINT `ads\_ibfk\_1` FOREIGN KEY (`author\_id`) REFERENCES `users` (`id`) ) ENGINE=InnoDB AUTO\_INCREMENT=42 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci

Таблица 2 — Характеристики полей таблицы Ads (Объявления)

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Свойства поля |
| id | INT | Ключевое поле |
| Счётчик |
| Обязательное поле |
| ads\_title | TEXT | Размер поля не органичен |
| Обязательное поле |
| ads\_text | TEXT | Размер поля не ограничен |
| Обязательное поле |
| ads\_wage | DECIMAL | Размер поля 8,2 |
| Обязательное поле |
| ads\_terms | TEXT | Размер поля не ограничен |
| Обязательное поле |
| ads\_visible | TINYINT | Значение 0 или 1 |
| ads\_likes | INT | Счётчик |
| ads\_create\_date | DATE | Обязательное поле |
| Выставляется автоматически |

Создание таблицы Driving (Опыт вождения), характеристики полей представлены в таблице 3.

CREATE TABLE `driving`(

`id` int(11) NOT NULL AUTO\_INCREMENT,

`driving\_name` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

PRIMARY KEY (`id`)

)ENGINE=InnoDB AUTO\_INCREMENT=12 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci

Таблица 3 — Характеристики полей таблицы Driving (Опыт вождения)

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Свойства поля |
| id | INT | Ключевое поле |
| Счётчик |
| Обязательное поле |
| driving\_name | TEXT | Размер поля не органичен |
| Обязательное поле |

Создание таблицы Education (Образование), характеристики полей представлены в таблице 4.

CREATE TABLE `education` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`education\_name` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

PRIMARY KEY (`id`)

)ENGINE=InnoDB AUTO\_INCREMENT=9 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci

Таблица 4 — Характеристики полей таблицы Education (Образование)

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Свойства поля |
| id | INT | Ключевое поле |
| Счётчик |
| Обязательное поле |
| education\_name | TEXT | Размер поля не органичен |
| Обязательное поле |

Создание таблицы Employment (Занятость), характеристики полей представлены в таблице 5.

CREATE TABLE `employment`(

`id` int(11) NOT NULL AUTO\_INCREMENT,

`employment\_name` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

PRIMARY KEY (`id`)

)ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci

Таблица 5 — Характеристики полей таблицы Employment (Занятость)

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Свойства поля |
| id | INT | Ключевое поле |
| Счётчик |
| Обязательное поле |
| employment\_name | TEXT | Размер поля не органичен |
| Обязательное поле |

Создание таблицы Experience (Опыт), характеристики полей представлены в таблице 6.

CREATE TABLE `experience`(

`id` int(11) NOT NULL AUTO\_INCREMENT,

`experience\_name` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

PRIMARY KEY (`id`)

)ENGINE=InnoDB AUTO\_INCREMENT=8 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci

Таблица 6 — Характеристики полей таблицы Experience (Опыт)

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Свойства поля |
| id | INT | Ключевое поле |
| Счётчик |
| Обязательное поле |
| experience\_name | TEXT | Размер поля не органичен |
| Обязательное поле |

Создание таблицы Languages (Языки), характеристики полей представлены в таблице 7.

CREATE TABLE `languages`(

`id` int(11) NOT NULL AUTO\_INCREMENT,

`language\_name` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

PRIMARY KEY (`id`)

)ENGINE=InnoDB AUTO\_INCREMENT=11 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci

Таблица 7 — Характеристики полей таблицы Languages (Языки)

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Свойства поля |
| id | INT | Ключевое поле |
| Счётчик |
| Обязательное поле |
| languages\_name | TEXT | Размер поля не органичен |
| Обязательное поле |

Создание таблицы Likes (Лайки), характеристики полей представлены в таблице 8.

CREATE TABLE `likes` (

`like\_id` int(11) NOT NULL AUTO\_INCREMENT,

`ad\_id` int(11) NOT NULL,

`user\_id` int(11) NOT NULL,

`like\_date` date NOT NULL,

PRIMARY KEY (`like\_id`),

KEY `ad\_id` (`ad\_id`,`user\_id`),

KEY `user\_id` (`user\_id`),

CONSTRAINT `likes\_ibfk\_1` FOREIGN KEY (`ad\_id`) REFERENCES `ads` (`ads\_id`),

CONSTRAINT `likes\_ibfk\_2` FOREIGN KEY (`user\_id`) REFERENCES `users` (`id`) ) ENGINE=InnoDB AUTO\_INCREMENT=33 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci

Таблица 8 — Характеристики полей таблицы Messages (Сообщения)

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Свойства поля |
| id | INT | Ключевое поле |
| Счётчик |
| Обязательное поле |
| ads\_id | INT | Размер поля не ограничен |
| Обязательное поле |
| Внешний ключ |

Продолжение таблицы 8 — Характеристики полей таблицы Messages (Сообщения)

|  |  |  |
| --- | --- | --- |
| user\_id | INT | Размер поля не ограничен |
| Обязательное поле |
| Внешний ключ |
| like\_date | DATE | Обязательное поле |
| Выставляется автоматически |

Создание таблицы Organization (Организация), характеристики полей представлены в таблице 9.

CREATE TABLE `organization`(

`id` int(11) NOT NULL AUTO\_INCREMENT,

`organization\_title` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

`organization\_description` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

`organization\_phone` varchar(11) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`organization\_email` varchar(64) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`organization\_adress` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

PRIMARY KEY (`id`)

)ENGINE=InnoDB AUTO\_INCREMENT=22 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci

Таблица 9 — Характеристики полей таблицы Organization (Организация)

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Свойства поля |
| id | INT | Ключевое поле |
| Счётчик |
| Обязательное поле |
| organization\_title | TEXT | Размер поля не ограничен |
| Обязательное поле |
| organization\_title | TEXT | Размер поля не органичен |
| Обязательное поле |
| organization\_description | TEXT | Размер поля не ограничен |
| Обязательное поле |

Продолжение таблицы 9 — Характеристики полей таблицы Organization (Организация)

|  |  |  |
| --- | --- | --- |
| organization\_phone | VARCHAR | Размер поля 11 |
| Обязательное поле |
| organization\_email | VARCHAR | Размер поля 64 |
| Обязательное поле |
| organization\_adress | TEXT | Размер поля не ограничен |
| Обязательное поле |

Создание таблицы Post (Должность), характеристики полей представлены в таблице 10.

CREATE TABLE `post`(

`id` int(11) NOT NULL AUTO\_INCREMENT,

`post\_name` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

PRIMARY KEY (`id`)

)ENGINE=InnoDB AUTO\_INCREMENT=31 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci

Таблица 10 — Характеристики полей таблицы Post (Должность)

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Свойства поля |
| id | INT | Ключевое поле |
| Счётчик |
| Обязательное поле |
| post \_name | TEXT | Размер поля не органичен |
| Обязательное поле |

Создание таблицы Responses (Сообщения к заявкам), характеристики полей представлены в таблице 11.

CREATE TABLE `responses` (

`responses\_id` int(11) NOT NULL AUTO\_INCREMENT,

`responses\_user\_id` int(11) NOT NULL,

`responses\_message` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

`responses\_ad\_id` int(11) NOT NULL,

`responses\_accepted` tinyint(1) NOT NULL,

`responses\_deny` tinyint(1) NOT NULL,

`responses\_date` date NOT NULL,

PRIMARY KEY (`responses\_id`),

KEY `responses\_user\_id` (`responses\_user\_id`,`responses\_ad\_id`),

KEY `responses\_ad\_id` (`responses\_ad\_id`),

CONSTRAINT `responses\_ibfk\_1` FOREIGN KEY (`responses\_ad\_id`) REFERENCES `ads` (`ads\_id`),

CONSTRAINT `responses\_ibfk\_2` FOREIGN KEY (`responses\_user\_id`) REFERENCES `users` (`id`) ) ENGINE=InnoDB AUTO\_INCREMENT=32 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci

Таблица 11 — Характеристики полей таблицы Responses (Сообщения к заявкам)

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Свойства поля |
| id | INT | Ключевое поле |
| Счётчик |
| Обязательное поле |
| responses\_user\_id | INT | Размер поля не ограничен |
| Обязательное поле |
| responses\_message | TEXT | Размер поля не органичен |
| Обязательное поле |
| responses\_ad\_id | INT | Размер поля не ограничен |
| Обязательное поле |
| Внешний ключ |
| responses\_accepted | TINYINT | Размер поля 1 |
| Обязательное поле |
| responses\_deny | TINYINT | Размер поля 1 |
| Обязательное поле |
| like\_date | DATE | Обязательное поле |
| Выставляется автоматически |

Создание таблицы Schedule (График), характеристики полей представлены в таблице 12.

CREATE TABLE ` schedule` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`schedule\_name` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

PRIMARY KEY (`id`)

)ENGINE=InnoDB AUTO\_INCREMENT=13 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci

Таблица 12 — Характеристики полей таблицы Schedule (График)

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Свойства поля |
| id | INT | Ключевое поле |
| Счётчик |
| Обязательное поле |
| schedule \_name | TEXT | Размер поля не органичен |
| Обязательное поле |

Создание таблицы Specialization (Специализация), характеристики полей представлены в таблице 13.

CREATE TABLE `specialization`(

`id` int(11) NOT NULL AUTO\_INCREMENT,

`specialization\_name` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

PRIMARY KEY (`id`)

)ENGINE=InnoDB AUTO\_INCREMENT=31 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci

Таблица 13 — Характеристики полей таблицы Specialization (Специализация)

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Свойства поля |
| id | INT | Ключевое поле |
| Счётчик |
| Обязательное поле |
| specialization\_name | TEXT | Размер поля не органичен |
| Обязательное поле |

Создание таблицы Summary (Резюме), характеристики полей представлены в таблице 14.

CREATE TABLE `summary` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`summary\_surname` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

`summary\_name` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

`summary\_patronymic` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

`summary\_age` int(11) NOT NULL,

`summary\_city` varchar(32) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`summary\_email` varchar(64) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`summary\_phone` varchar(11) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`experience\_id` int(11) NOT NULL,

`post\_id` int(11) NOT NULL,

`education\_id` int(11) NOT NULL,

`languages\_id` int(11) NOT NULL,

`specialization\_id` int(11) NOT NULL,

`driving\_id` int(11) NOT NULL,

`employment\_id` int(11) NOT NULL,

`schedule\_id` int(11) NOT NULL,

PRIMARY KEY (`id`),  KEY `experience\_id` (`experience\_id`,`post\_id`,`education\_id`,`languages\_id`,`specialization\_id`,`driving\_id`,`employment\_id`,`schedule\_id`),  KEY `driving\_id` (`driving\_id`),  KEY `education\_id` (`education\_id`),

KEY `employment\_id` (`employment\_id`),

KEY `languages\_id` (`languages\_id`),

KEY `post\_id` (`post\_id`),

KEY `schedule\_id` (`schedule\_id`),

KEY `specialization\_id` (`specialization\_id`),

CONSTRAINT `summary\_ibfk\_1` FOREIGN KEY (`driving\_id`) REFERENCES `driving` (`id`),

CONSTRAINT `summary\_ibfk\_2` FOREIGN KEY (`education\_id`) REFERENCES `education` (`id`),

CONSTRAINT `summary\_ibfk\_3` FOREIGN KEY (`employment\_id`) REFERENCES `employment` (`id`),

CONSTRAINT `summary\_ibfk\_4` FOREIGN KEY (`experience\_id`) REFERENCES `experience` (`id`),

CONSTRAINT `summary\_ibfk\_5` FOREIGN KEY (`languages\_id`) REFERENCES `languages` (`id`),

CONSTRAINT `summary\_ibfk\_6` FOREIGN KEY (`post\_id`) REFERENCES `post` (`id`),

CONSTRAINT `summary\_ibfk\_7` FOREIGN KEY (`schedule\_id`) REFERENCES `schedule` (`id`),

CONSTRAINT `summary\_ibfk\_8` FOREIGN KEY (`specialization\_id`) REFERENCES `specialization` (`id`) ) ENGINE=InnoDB AUTO\_INCREMENT=14 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci

Таблица 14 — Характеристики полей таблицы Summary (Резюме)

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Свойства поля |
| id | INT | Ключевое поле |
| Счётчик |
| Обязательное поле |
| summary\_surname | TEXT | Размер поля не органичен |
| Обязательное поле |
| summary\_name | INT | Размер поля не ограничен |
| Обязательное поле |
| summary\_patronymic | TEXT | Размер поля 1 |
| Обязательное поле |
| summary\_age | INT | Размер поля 1 |
| Обязательное поле |
| summary\_city | VARCHAR | Размер поля 32 |
| Обязательное поле |
| summary\_email | VARCHAR | Размер поля 64 |
| Обязательное поле |
| summary\_phone | VARCHAR | Размер поля 11 |
| Обязательное поле |
| experience\_id | INT | Размер поля не ограничен |
| Обязательное поле |
| Внешний ключ |
| post\_id | INT | Размер поля не ограничен |
| Обязательное поле |
| Внешний ключ |
| education\_id | INT | Размер поля не ограничен |
| Обязательное поле |
| Внешний ключ |
| languages\_id | INT | Размер поля не ограничен |
| Обязательное поле |
| Внешний ключ |
| specialization\_id | INT | Размер поля не ограничен |
| Обязательное поле |
| Внешний ключ |
| driving\_id | INT | Размер поля не ограничен |
| Обязательное поле |
| Внешний ключ |
| employment\_id | INT | Размер поля не ограничен |
| Обязательное поле |
| Внешний ключ |
| schedule\_id | INT | Размер поля не ограничен |
| Обязательное поле |
| Внешний ключ |

Создание таблицы Tages (Теги), характеристики полей представлены в таблице 15.

CREATE TABLE `tags` (

`id\_tag` int(11) NOT NULL AUTO\_INCREMENT,

`tag\_name` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

PRIMARY KEY (`id\_tag`)

) ENGINE=InnoDB AUTO\_INCREMENT=8 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci

Таблица 15 — Характеристики полей таблицы Tages (Теги)

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Свойства поля |
| id | INT | Ключевое поле |
| Счётчик |
| Обязательное поле |
| types \_name | TEXT | Размер поля не органичен |
| Обязательное поле |

Создание таблицы Tags\_for\_ads (Теги для объявлений), характеристики полей представлены в таблице 16.

CREATE TABLE `tags\_for\_ads` (

`id\_tfa` int(11) NOT NULL AUTO\_INCREMENT,

`tag\_id` int(11) NOT NULL,

`ad\_id` int(11) NOT NULL,

PRIMARY KEY (`id\_tfa`) ) ENGINE=InnoDB AUTO\_INCREMENT=68 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci

Таблица 16 — Характеристики полей таблицы Tags\_for\_ads (Теги для объявлений)

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Свойства поля |
| id | INT | Ключевое поле |
| Счётчик |
| Обязательное поле |
| tag\_id | INT | Внешний ключ |
| Обязательное поле |
| ad\_id | INT | Внешний ключ |
| Обязательное поле |

Создание таблицы Users (Пользователи), характеристики полей представлены в таблице 17.

CREATE TABLE `users`(

`id` int(11) NOT NULL AUTO\_INCREMENT,

`login` varchar(64) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`password` varchar(64) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`type\_id` tinyint(3) NOT NULL,

`summary\_id` int(11) NOT NULL,

`organization\_id` int(11) NOT NULL,

`blocked` tinyint(1) NOT NULL,

`deleted` tinyint(1) NOT NULL,

PRIMARY KEY (`id`)

)ENGINE=InnoDB AUTO\_INCREMENT=20 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci

Таблица 17 — Характеристики полей таблицы Users (Пользователи)

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Свойства поля |
| id | INT | Ключевое поле |
| Счётчик |
| Обязательное поле |
| login | VARCHAR | Размер поля 64 |
| Обязательное поле |
| password | VARCHAR | Размер поля 64 |
| Обязательное поле |
| type\_id | TINYINT | Размер поля 1 |
| Обязательное поле |
| Внешний ключ |
| summary\_id | INT | Размер поля не ограничен |
| Обязательное поле |
| Внешний ключ |
| organization\_id | INT | Размер поля не ограничен |
| Обязательное поле |
| Внешний ключ |
| blocked | TINYINT | Размер поля 1 |
| Обязательное поле |

Продолжение таблицы 17 — Характеристики полей таблицы Users (Пользователи)

|  |  |  |
| --- | --- | --- |
| deleted | TINYINT | Размер поля 1 |
| Обязательное поле |

## 2.5 Разработка физической и логической структуры сайта

Логическая структура сайта – это система ссылочного взаимодействия между страницами виртуального ресурса. То есть, взаимосвязь страниц, в которых прослеживается их принадлежность к разделам, категориям, подкатегориям, и другим типам страниц. Она должна продумываться на самых ранних этапах разработки сайта и является основной функциональности дизайна.

Структуру сайта надо построить таким образом, чтобы упростить пользователям переход от общих тем к конкретной информации, за которой они пришли. От этого зависит удобство пользователя, время, которое он проведет на сайте, целевые действия, которые он совершит.

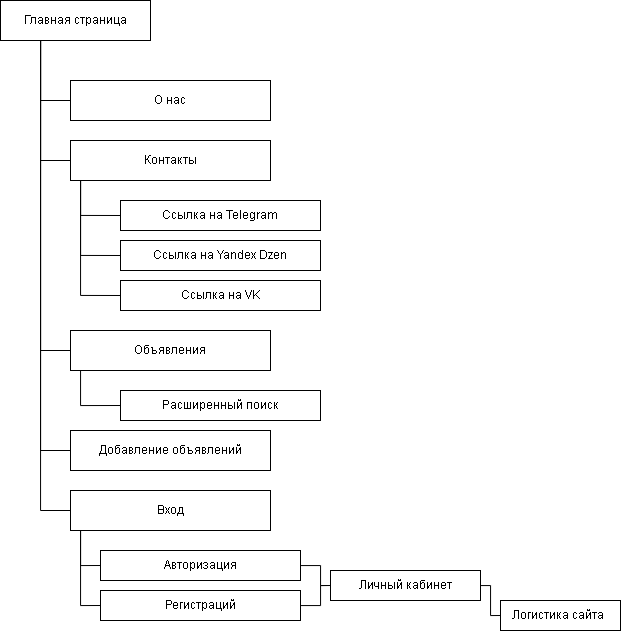


Рисунок 14 – Логическая структура сайта

Физическая структура сайта описывает размещение файлов и папок на диске компьютера. Правильная организация файлов и папок поможет в дальнейшей работе с сайтом.

Физическая и логическая структуры не всегда совпадают, потому что обычно физическая структура сайта разрабатывается, основываясь на удобстве размещения файлов. Но более или менее точное сохранение логических разделов в физической структуре сайта позволяет упростить работу по изменению обновлению информационных материалов.

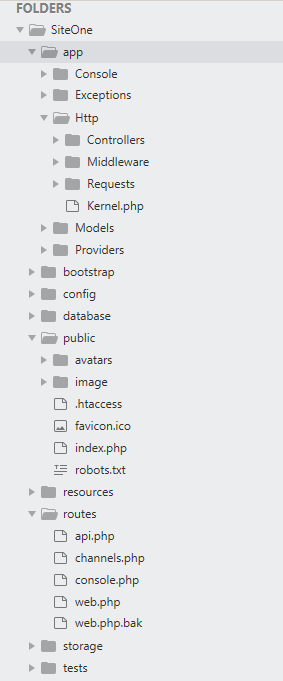


Рисунок 15 –Физическая структура сайта

## 2.6 Проектирование меню и пользовательского интерфейса

Проектирование пользовательского интерфейса – это создание тестовой версии приложения. Это начальный этап разработки пользовательского интерфейса, когда распределяются функции приложения по экранам, определяются макеты экранов, содержимое, элементы управления и их поведение.

Пользователь при обращении с интерфейсом должен представить себе, какая информация о выполняемой задаче у него существует, и в каком состоянии находятся средства, с помощью которых он будет решать данную задачу. Эффективность работы пользователя и его интерес обеспечивает правильно сформулированная методика разработки и проектирования пользовательского интерфейса.

Структура меню:

* о нас – ссылка на страницу, содержащая информацию о сайте;
* контакты – ссылка на страницу, содержащая контакты для связи с разработчиками;
* объявления – ссылка на страницу, содержащая объявление, которые загружены на сайт и содержит поиск и расширенный поиск;
* вход – ссылка на страницу, в которой можно зарегистрироваться или авторизоваться.

Главная страница предоставлена на рисунке 15.

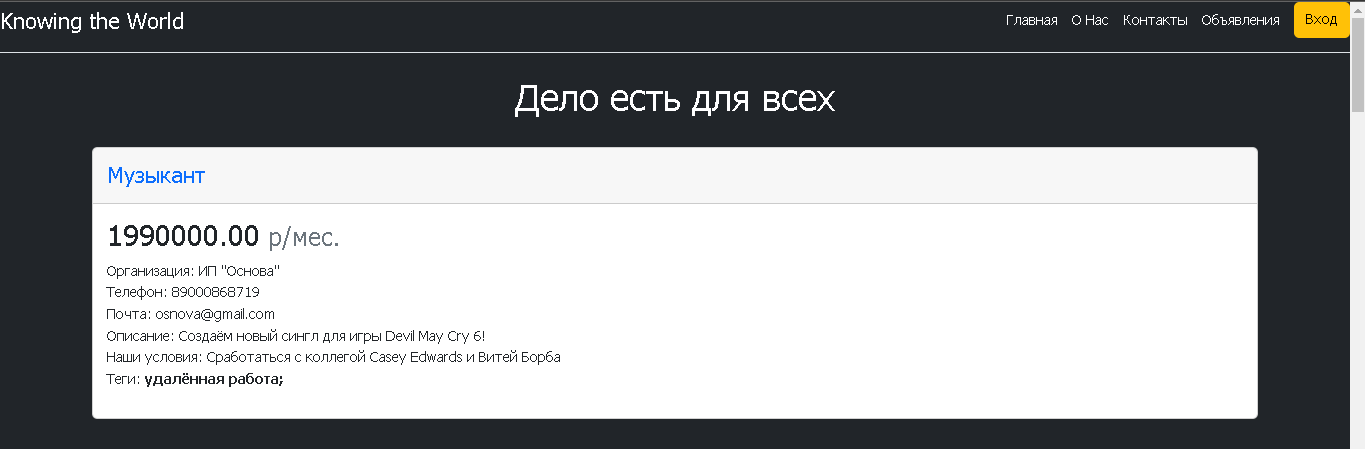


Рисунок 16 – Главная страница

На главной странице в шапке сайта с правой стороны расположено название сайта, а с левой стороны расположены ссылки на другие страницы сайта, а так же кнопка перехода на регистрацию/авторизацию.

Главная страница содержит объявления. Сначала идут пять случайных объявлений, после них пять объявлений, где больше всего лайков и завершается пятью объявлениями, где больше всего откликов на вакансию.

## 2.7 Описание запросов при разработке приложения

При разработке веб–приложения по поиску вакансий требуется выполнить следующие запросы:

Запросы для работы с пользователями:

* регистрация пользователей;
* авторизация пользователей;
* редактирование данных пользователя;
* добавление фотографии пользователя со стороны соискателя;
* создание объявлений со стороны работодателя.

Запросы для работы с объявлениями со стороны работодателя:

* создание объявлений;
* редактирование объявлений;
* скрытие объявления на сайте;
* просмотр сообщений к объявлениям;
* скачивание резюме пользователя в pdf формате;
* удаление объявления.

Запросы для работы с объявлениями со стороны соискателя:

* поставить отклик на объявление;
* отправить сообщение на объявление;
* поставить лайк на объявления.

Запросы для администратора:

* просмотр всех пользователей;
* скрытие объявлений на сайте;
* удаление объявлений на сайте;
* возможность заблокировать или разблокировать пользователей;
* просмотр графиков по определённым критериям.

Запросы должны быть эффективны в своём выполнении и выполнять высокую производительность веб–приложения. Результаты запросов должны быть представлены в скорректированном виде и обработаны на серверной стороне для безопасности пользователей и администратора.

Фрагмент кода при входе на сайт:

public function sign(){

$user = session('Пользователь');

$type = session('type');

$uid = session('uid');

if ($user==NULL) $user='';

if ($type==NULL) $type=–1;

if ($uid==NULL) $uid=–1;

if ($uid==NULL) $uid=–1;

$ads = array("" => "");

$resp = array("" => "");

$usrs = [];

$usrso = DB::select("SELECT \* FROM users");

if ($type==3){

$ads = DB::select("SELECT \* FROM ads");

$resp = DB::select("SELECT responses\_id, responses\_ad\_id , responses\_user\_id, responses\_accepted, responses\_deny, ads.ads\_title FROM responses INNER JOIN ads ON ads.ads\_id=responses.responses\_ad\_id");

foreach ($usrso as $u){

if ($u–>type\_id==1){

$ust = DB::select("SELECT \* FROM organization WHERE $u–>organization\_id=id");

foreach ($ust as $us){

array\_push($usrs,["id" => $u–>id, "name" => "$us–>organization\_title", "blocked" => $u–>blocked]);

}

}

else if ($u–>type\_id==2){

$ust = DB::select("SELECT \* FROM summary WHERE $u–>summary\_id=id");

foreach ($ust as $us){

array\_push($usrs,["id" => $u–>id, "name" => "$us–>summary\_name $us–>summary\_surname $us–>summary\_patronymic", "blocked" => $u–>blocked]);

}

}

}

return view('sign',['usrs' => $usrs, 'resp' => $resp,'ads' => $ads,'user' => $user, 'type' => $type, 'uid' => $uid]);

}

else {

$ads = DB::select("SELECT \* FROM ads WHERE author\_id=$uid");

$resp = DB::select("SELECT \* FROM responses INNER JOIN ads ON ads.ads\_id=responses.responses\_ad\_id INNER JOIN users ON users.id=responses\_user\_id INNER JOIN organization ON organization.id=users.organization\_id WHERE responses\_user\_id=$uid");

$experience = DB::select("SELECT \* FROM experience");

$post = DB::select("SELECT \* FROM post");

$education = DB::select("SELECT \* FROM education");

$language = DB::select("SELECT \* FROM languages");

$specialization = DB::select("SELECT \* FROM specialization");

$driving = DB::select("SELECT \* FROM driving");

$employment = DB::select("SELECT \* FROM employment");

$schedule = DB::select("SELECT \* FROM schedule");

$resume = DB::select("SELECT \*, users.id as uid FROM users INNER JOIN summary ON users.summary\_id=summary.id WHERE users.id=$uid");

return view('sign',['usrs' => $usrs, 'resp' => $resp,'ads' => $ads,'user' => $user, 'type' => $type, 'uid' => $uid, 'experience' => $experience

, 'post' => $post, 'education' => $education, 'language' => $language

, 'specialization' => $specialization, 'driving' => $driving

, 'employment' => $employment, 'schedule' => $schedule

, 'resume' => $resume]);

}

}

Фрагмент кода редактирование резюме соискателя.

$errors = "";

$user = session('Пользователь');

$type = session('type');

$uid = session('uid');

$experience = DB::select("SELECT \* FROM experience");

$post = DB::select("SELECT \* FROM post");

$education = DB::select("SELECT \* FROM education");

$language = DB::select("SELECT \* FROM languages");

$specialization = DB::select("SELECT \* FROM specialization");

$driving = DB::select("SELECT \* FROM driving");

$employment = DB::select("SELECT \* FROM employment");

$schedule = DB::select("SELECT \* FROM schedule");

$resume = DB::select("SELECT \* FROM users INNER JOIN summary ON users.summary\_id=summary.id WHERE users.id=$uid");

return view('edit\_resume',['user' => $user, 'type' => $type, 'uid' => $uid, 'experience' => $experience

, 'post' => $post, 'education' => $education, 'language' => $language

, 'specialization' => $specialization, 'driving' => $driving

, 'employment' => $employment, 'schedule' => $schedule

, 'resume' => $resume, 'errors' => $errors]);

Фрагмент кода регистрации.

$reg = $request->input('reg');

$errors = "";

$user = "";

$type = -1;

$uid = -1;

$arrayLogic = array('' => '');

if ($reg==0){

$arrayLogic = array('login' => ''

, 'password' => ''

, 'password\_1' => ''

, 'organization\_title' => ''

, 'organization\_description' => ''

, 'organization\_phone' => ''

, 'organization\_email' => ''

, 'organization\_adress' => ''

);

return view('registration',['user' => $user, 'type' => $type, 'uid' => $uid, 'reg' => $reg, 'arrayLogic' => $arrayLogic

, 'errors' => $errors]);

}

else {

$arrayLogic = array('login' => ''

, 'password' => ''

, 'password\_1' => ''

, 'summary\_surname' => ''

, 'summary\_name' => ''

, 'summary\_patronymic' => ''

, 'summary\_age' => ''

, 'summary\_city' => ''

, 'summary\_phone' => ''

, 'summary\_email' => ''

);

$experience = DB::select("SELECT \* FROM experience");

$post = DB::select("SELECT \* FROM post");

$education = DB::select("SELECT \* FROM education");

$language = DB::select("SELECT \* FROM languages");

$specialization = DB::select("SELECT \* FROM specialization");

$driving = DB::select("SELECT \* FROM driving");

$employment = DB::select("SELECT \* FROM employment");

$schedule = DB::select("SELECT \* FROM schedule");

return view('registration',['user' => $user, 'type' => $type, 'uid' => $uid, 'reg' => $reg, 'experience' => $experience

, 'post' => $post, 'education' => $education, 'language' => $language

, 'specialization' => $specialization, 'driving' => $driving

, 'employment' => $employment, 'schedule' => $schedule

, 'arrayLogic' => $arrayLogic, 'errors' => $errors]);

}

Фрагмент кода расширенного поиска

$user = session('Пользователь');

$type = session('type');

$uid = session('uid');

if ($uid==NULL) $uid=-1;

$name\_ad = $request->input('name\_ad');

$name\_org = $request->input('name\_org');

$wage1 = $request->input('wage1');

$wage2 = $request->input('wage2');

$tag = $request->input('tag');

$sort = $request->input('sort');

DB::insert("INSERT INTO searches(search\_id, search\_body, search\_date) VALUES(NULL,'$name\_ad',NOW())");

$tfa = DB::select("SELECT \* FROM tags\_for\_ads");

$tags = DB::select("SELECT \* FROM tags");

if (($name\_ad=='') and ($name\_org=='')

and ($wage1=='') and ($wage2=='') and ($tag=='Нет'))

{

$ads = DB::select("SELECT \* FROM ads INNER JOIN users ON ads.author\_id=users.id INNER JOIN organization ON users.organization\_id=organization.id ORDER BY ads.ads\_title");

$errors = "";

$res\_id = DB::select("SELECT responses\_id, responses\_user\_id, responses\_ad\_id FROM responses;");

return view('news', ['res\_id' => $res\_id, 'errors' => $errors, 'ads' => $ads, 'user' => $user, 'type' => $type, 'uid' => $uid, 'tags' => $tags,'tfa' => $tfa]);

}

$s = "SELECT \* FROM ads INNER JOIN users ON ads.author\_id=users.id INNER JOIN organization ON users.organization\_id=organization.id";

if ($name\_ad!='') $s = $s." WHERE ads.ads\_title LIKE "."'%$name\_ad%'";

if ($name\_org!='')

{

if ($name\_ad!='') $s = $s." AND organization.organization\_title LIKE '%$name\_org%'";

else $s = $s." WHERE organization.organization\_title LIKE '%$name\_org%'";

}

if ($wage1!='')

{

if (($name\_ad!='') or ($name\_org!='')) $s = $s." AND ads.ads\_wage >= $wage1";

else $s = $s." WHERE ads.ads\_wage >= $wage1";

}

if ($wage2!='')

{

if (($name\_ad!='') or ($name\_org!='') or ($wage1!='')) $s = $s." AND ads.ads\_wage <= $wage2";

else $s = $s." WHERE ads.ads\_wage <= $wage2";

}

if ($sort==0)

{

$s = $s." ORDER BY ads.ads\_title ASC;";

}

else if ($sort==1)

{

$s = $s." ORDER BY ads.ads\_title DESC;";

}

else if ($sort==2)

{

$s = $s." ORDER BY ads.ads\_likes ASC;";

}

else if ($sort==3)

{

$s = $s." ORDER BY ads.ads\_likes DESC;";

}

else if ($sort==4)

{

$s = $s." ORDER BY ads.ads\_responses ASC;";

}

else if ($sort==5)

{

$s = $s." ORDER BY ads.ads\_responses DESC;";

}

$ads = DB::select($s);

$errors = "";

$res\_id = DB::select("SELECT responses\_id, responses\_user\_id, responses\_ad\_id FROM responses;");

return view('news', [ 'res\_id' => $res\_id, 'errors' => $errors, 'ads' => $ads, 'user' => $user, 'type' => $type, 'uid' => $uid, 'tags' => $tags,'tfa' => $tfa,'tag'=>$tag]);

Фрагмент кода добавление изображения профиля

$request->validate([

'image' => 'required|image|mimes:jpeg,png,jpg,gif,svg|max:10000',

]);

$uid = $request->input('uid');

$imageName = $uid.time().'.'.$request->image->extension();

$request->image->move(public\_path('avatars'), $imageName);

$path = "avatars/".$imageName;

DB::update("UPDATE users SET avatar='$path' WHERE id=$uid");

return redirect('/sign');

## 2.8 Проектирование отчетов

Входной документы – это документы, содержащие исходную запись сведений, полученных в процессе исследований, разработок, наблюдений, анализа или других видов человеческой деятельности независимо от ее характера или тематики. Деление документов на первичные и вторичные в значительной степени условно, поскольку один и тот же документ может содержать сведения, относящиеся как к первой, так и второй группе. Так, считающиеся безусловно первичными видами документов – отчеты по НИР, проектные документы, монографии и т.д., могут содержать наряду с оригинальными сведения или данные ранее включенные в другие первичные или даже вторичные документы.

Выходной документы – это документы (отчеты), полученные в результате аналитико–синтетической и логической переработки сведений или данных, содержащихся в первичных документах. Примерами вторичных документов являются справочные и энциклопедические издания, рефераты и реферативные издания, библиографические издания, указатели и списки, обзоры (за исключением т.н. аналитических обзоров) и т.п.

К выходным документам относится: отчет о поступлении товара, отчет о продажах, о поступлении денежных средств.

Входные и выходные документы должны, как правило, выполняться на типовых, специализированных и стандартных бланках, что позволяет использовать для их заполнения.

Формат PDF — расшифровывается как мобильный формат документов (Portable Document Format). Он является потомком языка PostScript (профессиональной надстройкой над ним), созданного фирмой Adobe Systems. Формат PDF позволяет сохранять в одном документе фотографии, векторную графику, текст, при различной сложности вёрстки. Документ PDF поддерживает кроссплатформенность, сохраняя при этом точность отображения или печати страницы, вплоть до мельчайших деталей — пробела, точки, знака, расстояний.

PDF — это удобный формат для публикации страниц и информации в Интернете. Он прекрасно индексируется поисковыми системами, может включать несколько уровней защиты от копирования, печати, внесения изменений и добавления веб–форм.

Выходным документом данного веб–приложения является резюме соискателя, который скачивает работодатель после одобрения отклика на выборанную соискателем вакансию. Резюме представляет из себя pdf файл, который можно скачать. Содержимое документа представляет из себя информацию о соискателе, а так же его контактных данных.



Рисунок 17 – Создания объявления

## 2.9 Администрирование программного продукта

Администрирование программного обеспечения - это приведение программного обеспечения в соответствие с целями и задачами, для которых оно предназначено. Достигается путём управления, позволяющего минимизировать затраты времени и ресурсов, направляемых на управление системой, и в тоже время максимизировать доступность, производительность и продуктивность системы.

Возможность удаленного выполнения указанных функций позволяет заменить штатного системного администратора на систему удаленного администрирования и существенно сократить расходы.

Принципы администрирования проекта

* мониторинг процессов программных комплексов и серверных систем;
* оперативное исправление ошибок и сбоев в работе программных комплексов и серверных систем;
* совершенствование программных комплексов, обновление систем, анализ и установка новых перспективных модулей.

Администрирование базы данных подразумевает под собой разграничение ролей пользователя. Оно необходимо для сохранности и конфиденциальности информации.

Рассмотреть администрирование программного обеспечения, описать категории пользователей, доступный им функционал приложения, а также накладываемые ограничения (рис. 19).

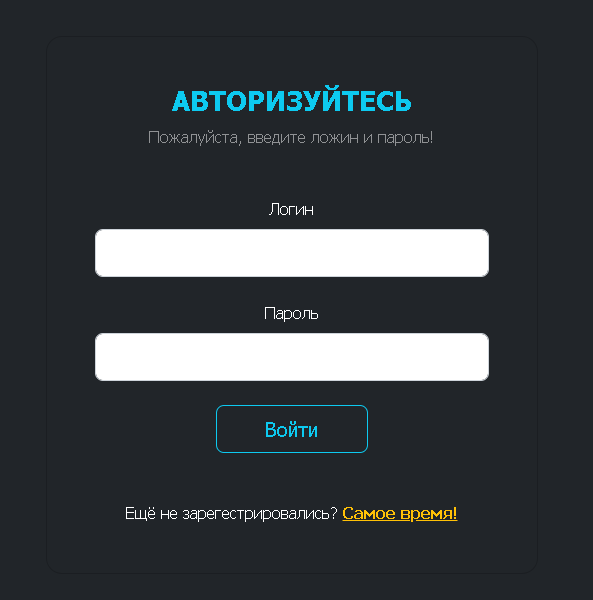


Рисунок 19 — Авторизация

# **3 ОПИСАНИЕ РАБОТЫ ПРОГРАММНОГО ПРОДУКТА**

## 3.1 Руководство пользователя

Работа с сайтом начинается с его запуска. На сайте можно перейти по страницам «О нас», «Контакты», «Объявления» и «Вход». На странице «Вход» происходит авторизация пользователя.

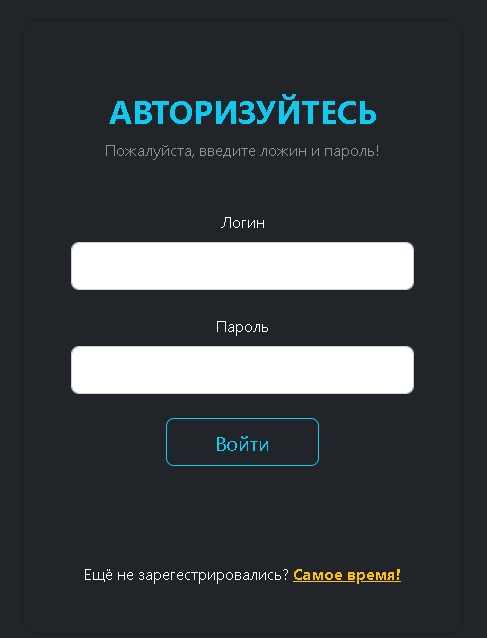


Рисунок 20 — Авторизация

Если пользователь уже зарегистрирован, то он переходит в свой личный кабинет. Если же пользователь является незарегистрированным, то переходя по ссылке на регистрацию, будет предоставлен выбор, какую роль может выбрать пользователь.

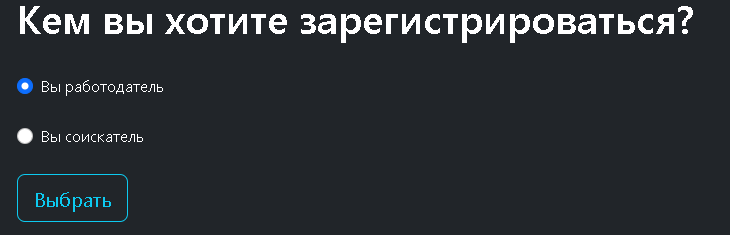


Рисунок 21 — Выбор типа регистрации

Если пользователь выбрал зарегистрироваться как работодатель, то ему будет предоставленная следующая форма регистрации.

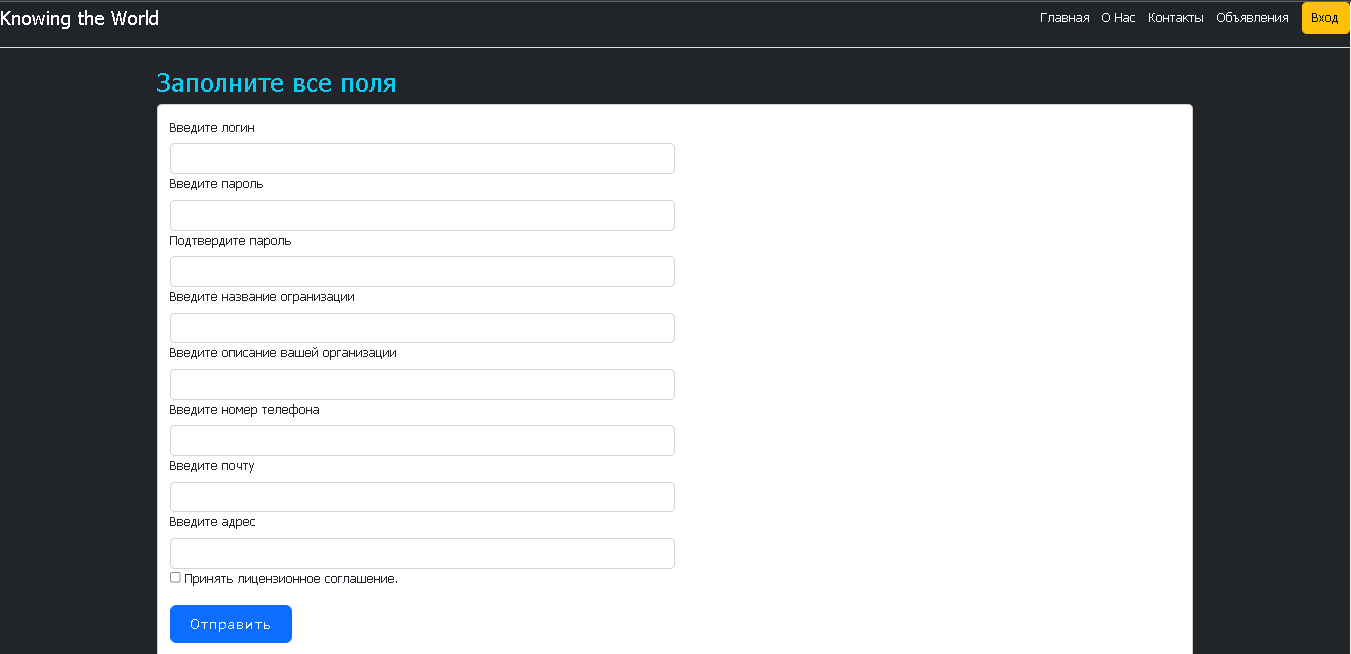


Рисунок 22 — Форма регистрации работодателя

В случае если пользователь хочет зарегистрироваться как соискатель, ему будет представлена другая форма регистрации.

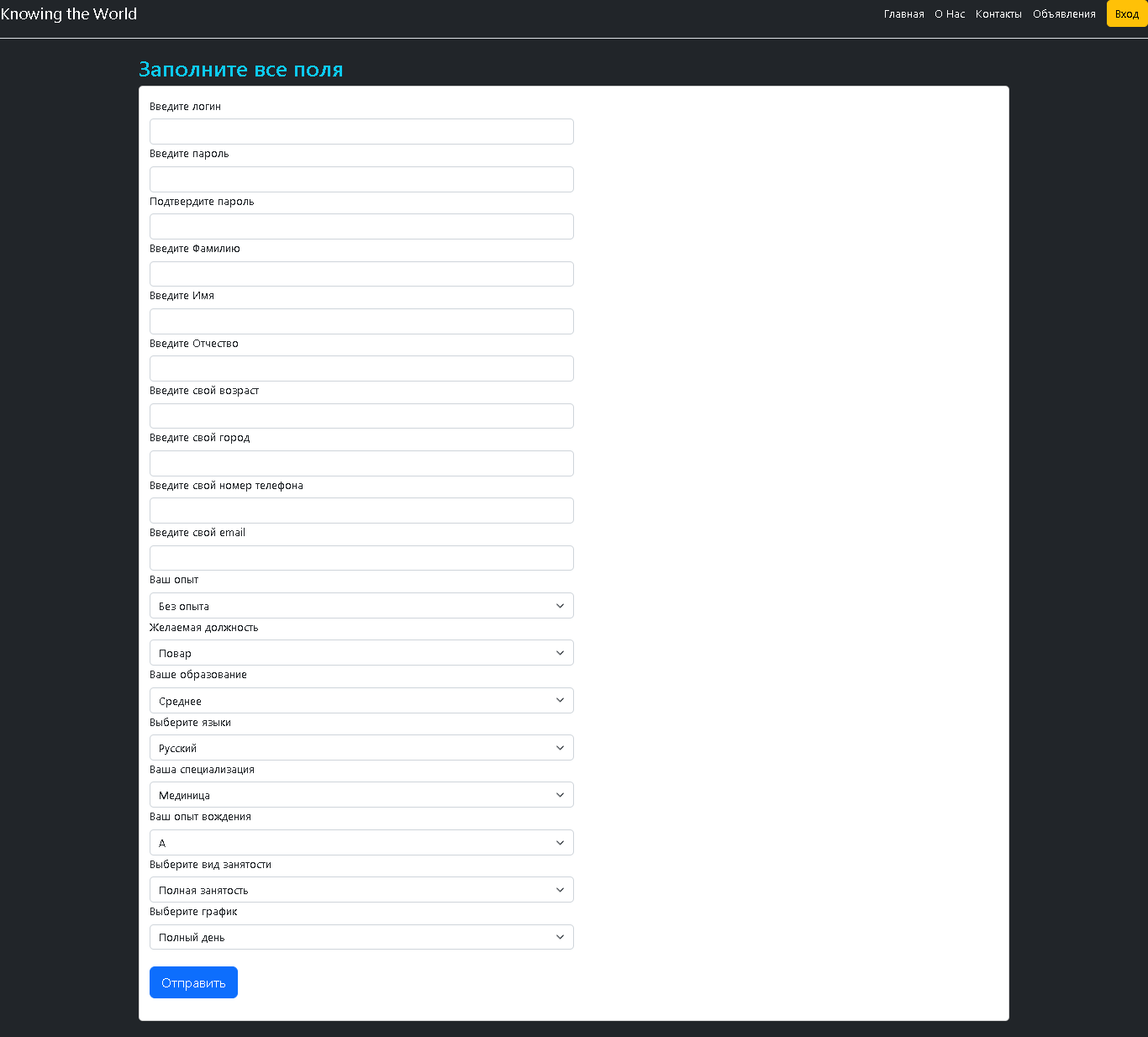


Рисунок 23 — Форма регистрации соискателя

После регистрации пользователя возвращают на страницу авторизации, где он вводит свой созданный логин и пароль. После удачного входа пользователь отправляется на страницу личного кабинета. На этой странице работодателю предоставляются все его объявления, которые он создал. На этой странице работодатель может редактировать своё объявление, скрыть его на сайте, удалить, а так же посмотреть сообщения оставленные соискателями на данное объявление.

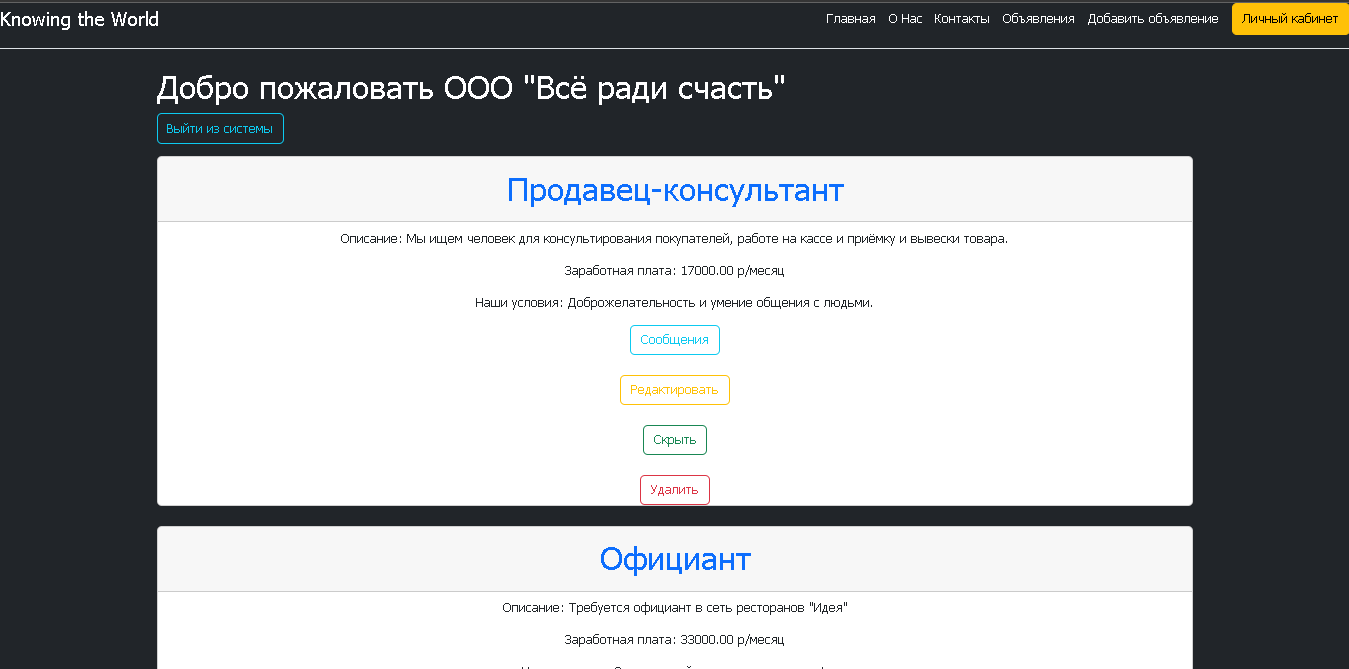


Рисунок 24 — Личный кабинет работодателя

Страница сообщения содержит информацию от соискателя, который откликнулся на объявление. Здесь предоставлена информация о соискателе, его резюме, фотография пользователя и сообщение, написанное к объявлению. Работодатель может как подтвердить отклик пользователя, так и отклонить его.

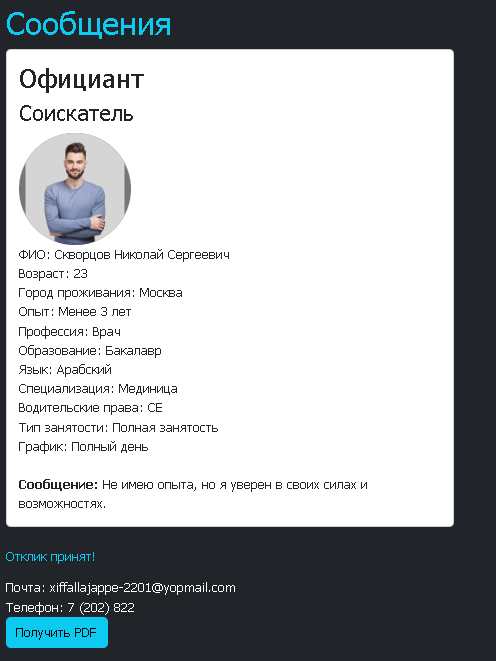


Рисунок 25 — Сообщение к объявлению от соискателя

Если работодатель одобрил отклик, он получает так же его адрес и номер телефона. После одобрения отклика так же доступна функция скачать резюме соискателя в PDF формате.



Рисунок 26 — Скачанное резюме соискателя

Функционал соискателя сильно отличается от возможностей работодателя. Соискатель при просмотре объявлений может оставить лайк на понравившуюся вакансию, а так же отправить отклик с сообщением работодателю.

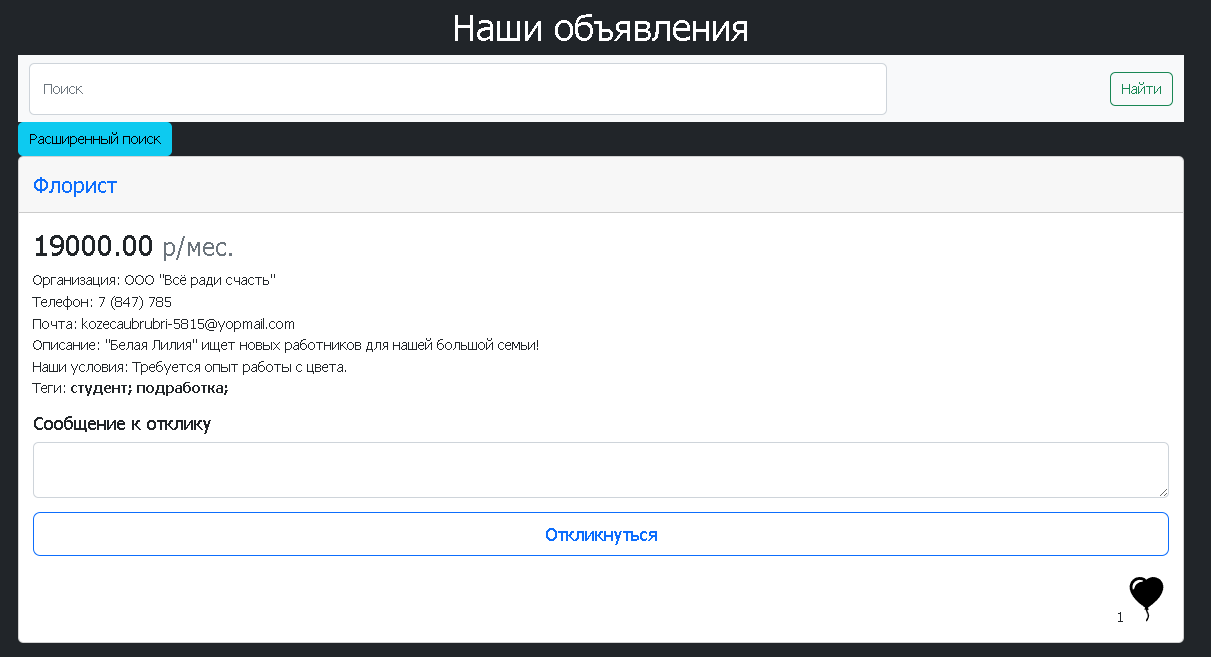


Рисунок 27 — Страница с объявлениями со стороны соискателя

После отправления отклика на вакансию, возможность откликнуться ещё раз пропадает. После, в зависимости от решения работодателя, соискатель получит сообщение в личном кабинете либо о принятии его отклика, либо о его отклонении.

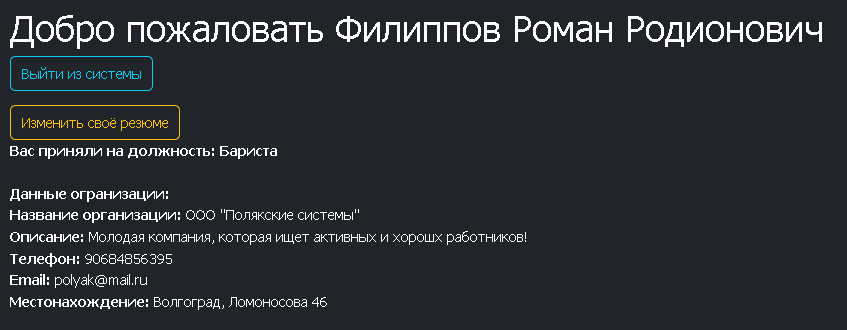


Рисунок 28 — Личный кабинет с сообщение на отклик

## 3.2 Руководство программисту

Сайт по поиску вакансий создан для большого объема хранимых в нём данных.

На странице главной должны выводиться объявления. Вначале идут пять рандомных объявлений. Случайные объявления не должны быть со статусом «скрыто». Далее идут пять объявлений, где стоит больше всего лайков. Завершается всё это объявлениями, куда больше всего отправлено откликов от соискателей.

В зависимости от того, является пользователем, зарегистрированным или незарегистрированным, будет представлены объявления на сайте. Если пользователь незарегистрированный, то ему будет предоставлено объявление, на которое нельзя откликнуться или поставить лайк.

На страницах «объявления» содержаться все объявления, которые есть в базе данных, а так же поисковая строка. Порядок вывода начинается от даты добавления на сайт по убыванию. Под поисковой строкой расположена кнопка перехода на расширенный поиск.

Страница с расширенным поиском содержит поисковые строки. Поиск может осуществляться по наименованию заявки, по наименованию организации, по желаемой заработной плате в выбранном диапазоне или по тегам. Так же можно применить сортировку найденных объявлений по алфавиту от а до я, или наоборот. Или по лайкам по возрастанию или убыванию. Та же схема, применимая в лайках осуществляется и в сортировке по датам добавления объявлений.

Регистрация пользователей различается от выбранной пользователем роли. Так же и с личным кабинетом. Страница с личным кабинетом для соискателя показывает сообщения на оставленные отклики. Сообщение состоит из информации о вакансии, а так же информацию об организации. Ниже идёт резюме соискателя, которое он создавал при регистрации. Так же там расположена фотография соискателя и кнопка обновления фотографии. Выше резюме и сообщение на отклик находиться кнопка выхода из системы и кнопка редактирование резюме соискателя.

Личный кабинет работодателя представляет из себя страницу со всеми объявлениями, которые выкладывал данный работодатель. В самом объявлении вместо кнопки отклика расположены кнопки редактирования, скрытия, удаления и просмотр сообщений к отклику. Кнопка просмотра сообщений переводят на страницу, где показана страница с резюме от соискателя и его сообщение, написанное к вакансии. Так же там расположены кнопки принятия или отказа от отклика соискателя. При принятии отклика появляется возможность скачать резюме соискателя в pdf формате.

Кабинет администратора содержит всё объявления, которые есть в базе данных. Он может их удалять или скрывать. Так же администратор видит всех пользователей, и может их заблокировать или разблокировать. По кнопке «логистика» находятся графики, которые строятся в зависимости от выбора и сортировки.

Список сортировок;

* статистика регистрации соискателя за период;
* статистика регистрации работодателя за период;
* статистика заблокированных соискателей за период;
* статистика объявлений за период;
* статистика объявлений за период;
* статистика удалённых объявлений за период;
* статистика откликов за период;
* статистика лайков за период.

Возможные выборы периода:

* за всё время;
* за месяц;
* за неделю.

База данных содержит сведения о созданных объявлениях, резюме соискателей, информации о сообщениях, отправленных к отклику и так далее. Пароли пользователей поддаются хешированию в базе данных. Фотографии, которые соискатель ставит в свой профиль могут иметь разные форматы файла. База данных выполнена в программе PhpMyAdmin.

Для работы сайта на локальный сервер приведены настройки в таблице 18.

Таблица 18 — Настройки для локального сервера

|  |  |
| --- | --- |
| HTTP | Apache2.4 — PHP8.0—8.1 |
| PHP | 8.0 |
| MariaDB | MySQL 8.0 |
| PORT | 3306 |

При успешной установке настроек требуется проверить работоспособность веб–приложения. Провести проверку всего функционала через соискателей и работодателей. Добавить объявление и проверить его корректное добавление в базу данных.

Так же стоит провести исправность работы панели администратора. В панели проверяются возможность удаление и скрытие объявлений, возможность заблокировать и разблокировать пользователей, а так же работа графиков для статистики.

## 3.3 Руководство администратора

Страница личного кабинета администратора. Содержит все объявления и список всех пользователей. На данной странице можно удалить или скрыть объявления. Администратор может заблокировать или разблокировать соискателей и/или работодателей.

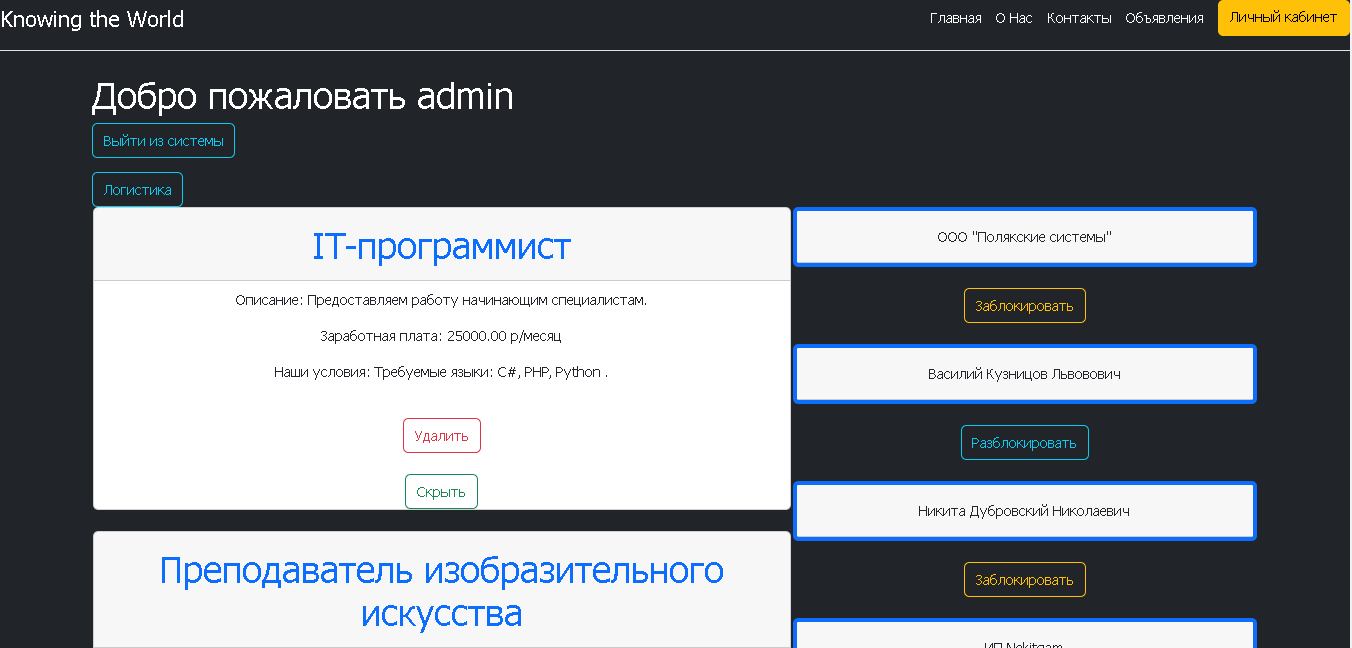


Рисунок 18 – Личный кабинет администратора

Администратор может посмотреть анализ работы сайта на странице Логистика. Данная страница содержит возможные категории для просмотра активности сайта. Статистика может выводиться за определённые периоды, такие как:

* за всё время;
* за месяц;
* за неделю.

Вывод статистики происходит в виде графика.

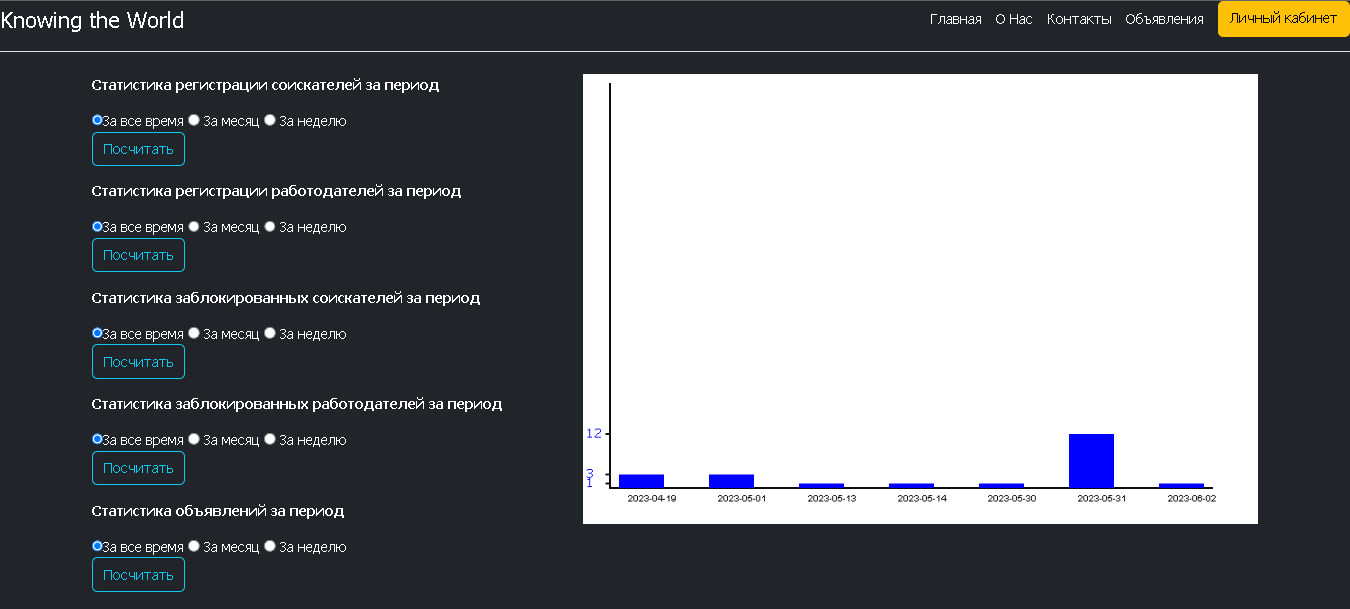


Рисунок 19 – График лайков за всё время

# **4 ИНФОРМАЦИОННАЯ БЕЗОПАСНОСТЬ**

## 4.1 Общие положение информационной безопасности, уровни информационной безопасности

Информационная безопасность – это сохранение и защита информации, а также ее важнейших элементов, в том числе системы и оборудование, предназначенные для использования, сбережения и передачи этой информации. Иначе, это набор технологий, стандартов и методов управления, которые необходимы для защиты информационной безопасности.

Главной целью информационной безопасности является защита информационных данных и поддерживание инфраструктуру от случайного или специального вмешательства.

Информационная безопасность придерживается трех главных принципов:

* конфиденциальность. Это гарантирует достаточный уровень безопасности с данными пользователей, которые пользуются веб–приложением;
* целостность. Она обеспечивает предотвращение искажения информации;
* доступность. Доступность обеспечивает надежный и эффективный доступ к информации уполномоченных лиц.

Угроза объекту информационной безопасности есть совокупность факторов и условий, возникающих в процессе взаимодействия различных объектов (их элементов) и способных оказывать негативное воздействие на конкретный объек информационной безопасности. Негативные воздействия различаются по характеру наносимго вреда, а именно: по степени изменения свойств объекта безопасности и возможности ликвидации последствий проявления угрозы.

Выделяется несколько уровней информационной безопасности:

* законодательный. Он включается в себя совокупность мер, которые направлены на формирование и поддержку в обществе отрицательного отношения к киберприступникам и нарушителям информационной безопасности. Так же включается в себя подготовку отечественных норм и стандартов в соответствие с международными регламентами.
* административный. Административной безопасностью занимаются руководство разработчиков. Главный документ на данном уровне – политика безопасности, состоящая из решений по управлению, необходимых для защиты данных, техсредств, которые с ней ассоциированы.
* процедурный. Сюда входят некоторые меры безопасности, такие как обеспечение физической защиты техсредств, программ, документации, поддержка работоспособности, реагирование на негативные события в режиме безопасности и стратегия восстановительных работ.
* программно–технический уровень. Здесь используются основные механизмы безопасности, такие как обеспечение максимальной доступности, протоколирование, управление доступом, идентификация и проверка подлинности пользователей.

В России действуют законы, где описано, как правильно работать с информацией: кто отвечает за ее сохранность, как ее собирать, обрабатывать, хранить и распространять.

Федеральный закон "Об информации, информационных технологиях и о защите информации" является главный закон об информации в России. Он определяет ключевые термины, например, говорит, что информация — это любые данные, сведения и сообщения, представляемые в любой форме. Также там описано, что такое сайт, электронное сообщение и поисковая система. Именно на этот закон и эти определения нужно ссылаться при составлении документов по информационной безопасности.

В 149 федеральном законе сказано, какая информация считается конфиденциальной, а какая – общедоступной, когда и как можно ограничивать доступ к информации, как происходит обмен данными. Также именно здесь прописаны основные требования к защите информации и ответственность за нарушения при работе с ней.

Ключевые моменты закона об информационной безопасности:

* нельзя собирать и распространять информацию о жизни человека без его согласия;
* все информационные технологии равнозначны — нельзя обязать компанию использовать какие-то конкретные технологии для создания информационной системы;
* есть информация, к которой нельзя ограничивать доступ, например сведения о состоянии окружающей среды;
* некоторую информацию распространять запрещено, например ту, которая пропагандирует насилие или нетерпимость;
* тот, кто хранит информацию, обязан ее защищать, например, предотвращать доступ к ней третьих лиц.

Федеральный закон «О персональных данных» регулирует работу с персональными данными – личными данными конкретных людей. Его обязаны соблюдать те, кто собирает и хранит эти данные.

Ключевые моменты закона:

* перед сбором и обработкой персональных данных нужно спрашивать согласие их владельца;
* для защиты информации закон обязывает собирать персональные данные только с конкретной целью;
* если вы собираете персональные данные, то обязаны держать их в секрете и защищать от посторонних;
* если владелец персональных данных потребует их удалить, вы обязаны сразу же это сделать;
* если вы работаете с персональными данными, то обязаны хранить и обрабатывать их в базах на территории Российской Федерации. При этом данные можно передавать за границу при соблюдении определенных условий, прописанных в законе — жесткого запрета на трансграничную передачу данных нет.

Федеральный закон "О безопасности" от 28.12.2010 N 390 определяет основные принципы и содержание деятельности по обеспечению безопасности государства, общественной безопасности, экологической безопасности, безопасности личности, иных видов безопасности, предусмотренных законодательством Российской Федерации (далее - безопасность, национальная безопасность), полномочия и функции федеральных органов государственной власти, органов государственной власти субъектов Российской Федерации, органов местного самоуправления в области безопасности, а также статус Совета Безопасности Российской Федерации.

## Федеральный закон 187 «О безопасности критической информационной инфраструктуры Российской Федерации»

Этот закон касается компаний, которые работают в сферах, критически важных для жизни государства — таких, что сбой в их работе отразится на здоровье, безопасности и комфорте граждан России.

К таким сферам относится здравоохранение, наука, транспорт, связь, энергетика, банки, топливная промышленность, атомная энергетика, оборонная промышленность, ракетно-космическая промышленность, горнодобывающая промышленность, металлургическая промышленность и химическая промышленность. Также сюда относят компании, которые обеспечивают работу предприятий из этих сфер, например, предоставляют оборудование в аренду или разрабатывают для них программное обеспечение.

Если на предприятии из этой сферы будет простой, это негативно отразится на жизни всего государства. Поэтому к IT-инфраструктуре и безопасности информационных систем на этих предприятиях предъявляют особые требования.

## 4.2 Виды возможных угроз веб–приложения по поиску вакансий

Программное обеспечение веб–приложения по поиску вакансий может подвергнуться различными видами угроз безопасности, которые могут привести к краже или уничтожению конфиденциальной информации пользователей и администратора.

Самые распространённые виды угроз для веб–приложения по поиску вакансий состоит из:

* XSS—атаки – распространённая уязвимость, когда злоумышленники внедряют на страницу JavaScript–код, которые не был предусмотрен. Этот код будет выполнятся каждый раз, когда пользователи будут заходить на страницу с вредоносным кодом;
* Сlickjacking позволяет вредоносной странице кликнуть по взломанному сайта от имени пользователя;
* DoS–атака – атака, которая старается довести атакуемый сервис до отказа. Это может работать так, что пользователь не смог бы открыть страниц веб–приложения или совершить какой–либо функционал;
* SQL–инъекции – их суть заключается в доступе к данным атакуемого сервиса или их изменению, или даже удалению. Злоумышленники будет изменять и добалять собственный SQL–запрос;
* Key logger – атака, которая состоит из того, что злоумышленники создали программу которая отправляет все вводимые пользователем данные. Таким образом у пользователей могут украсть пароли и другие персональные данные.

Защита информации которая есть в веб–приложении, такие как личные данные, информация с сайта и данные администратора должны быть в безопасности. Выполняя определённые меры для обеспечения информационной безопасности этого можно избежать.

## 4.3 Информационная безопасность программного продукта

Для безопасности веб-приложения была проведена проверка вводимых данных со стороны пользователя. Это позволяет предотвратить неправильные или нежеланные вводимые данные, включая те, которые могут быть использованы злоумышленниками для SQL, CSRF, других атак.

Так же было использовано хеширование паролей. Все пароли пользователей будут хешироваться и храниться в зашифрованном виде, который предотвращает кражу данных со стороны злоумышленников.

Так же были использованы подготовленные запросы, которые позволяют предотвратить SQL-инъекции, поскольку запросы выполняются отдельно от данных, вводимых пользователем.

При этом были использованы защищённые протоколы передачи данных, которые защищают пользовательские сессии от несанкционированного доступа к данным.

Вместе с тем производиться регулярное резервное копирование веб-приложения и всех важных данных. И так же проводиться тестирование веб-приложения, которое позволяет убедиться, что не были внедрены вредоносные ссылки.

# **5 УПРАВЛЕНИЕ ПРОЕКТОМ**

## 5.1 Понятие проекта. Необходимость управления проектом

Проект веб–приложения по поиску вакансий – это совокупность действий, ограниченных во времени и направленных на решение проблемы или достижение конкретной цели. Управление проектом веб–приложения по поиску вакансий – это процесс, который позволяет достигать поставленной цели, обозначить действия, просмотреть возможные риски и организовать ресурсы для повышения эффективности проекта.

Управление проектом как раз помогает быстро и эффективно достигать заданных целей. Кроме того, в процессе этого формируется целая система комплексов, которые могут быть задействованы для общих целей при создании веб–приложения, и разрабатывается схема грамотного распределения ресурсов.

В систему управления проекта входит ряд последовательных действий:

* определение и формирование требований к проекту;
* постановка максимально четких и понятных целей;
* установка и реализация коммуникации между задействованными в проекте сторонами;
* уравновешивание проектных ограничений: в частности бюджета, ресурсов, рисков, дедлайнов, качества.

Процесс управления проектом состоит в основном из пяти основных стадий проекта:

* инициирование проекта. На стадии инициирования оценивается его масштабность. В зависимости от сложности и объёма проекта также появляется возможность составить дорожну карту;
* планирование проекта. При планировании проекта определяется требования проекта и то, как будет выглядеть его «успех». Данная стадия очень важна для качественного управления проектом и достижения поставленных в нём целей. На этом этапе создаётся план проекта и идёт получение представления о распределении ресурсов в хронологическом порядке;
* выполнение проекта. Является самой длительной стадией проекта. На данной стадии рекомендуется принять методы управление загрузкой и управление задачами, чтобы синхронизировать работу создания проекта;
* отчётность по проекту. В ходе работы над проектом отчётность поможет корректировать курс, налаживать работу и делать информацию доступной для понимая. После завершении работы над проектом можно составить отчёт о проделанной работе;
* закрытие проекта. После завершения работы над проектом стоит произвести анализ выполненной работы, чтобы зафиксировать данные и вынести урок из полученного итога.

## 5.2 Постановка цели проекта по SMART

SMART – это метод создания тщательно спланированной, ясной и отслеживаемой цели. Он помогает расписать цель по пунктам, чтобы сосредоточить усилия на конкретных шагах и обеспечить качественный результат.

Цель проекта веб–приложения по поиску вакансий была поставлена по SMART, чтобы внести конкретность и ясность и показать актуальность веб–приложения.

Таблица 19 — Постановка цели проекта по SMART

|  |  |  |
| --- | --- | --- |
| Обозначение | Название | Значение |
| S | Specific (Конкретный) | Создание веб–приложения по поиску вакансий |
| M | Measurable (Измеримый) | Обеспечить пригодность к эксплуатации в указанные сроки |
| A | Attainable (Достижимый) | Исходя из общедоступных ресурсов и сроков реализации проекта создать готовое веб–приложение по поиску вакансий |
| R | Relevant (Актуальный) | Разработать веб–приложение основываясь на современных и популярных источниках. |
| T | Time–bound (Ограниченный по времени) | Разработать готовое веб–приложение в указанные сроки |

Таким образом постановка SMART для веб–приложения показывает ясность цели, определит её актуальность и достижимость, и организацию сроков для создания качественного веб–приложения.

## 5.3 Анализ заинтересованных сторон проекта

Анализ заинтересованных сторон проекта позволяет охарактеризовать заинтересованные стороны, а также оценить их интерес к проекту. Это является действенным инструментом управления, так как он дает возможность задавать стратегические вопросы, помогает понять сложность рассматриваемого вопроса и представит в результате перечини основных характеристик каждой из заинтересованных сторон.

* соискатели – самая заинтересованная сторона, так как они будут использовать веб–приложение для поиска требуемой им работы и изучения рынка труда;
* работодатели – так же являются одной из главных заинтересованных сторон, ведь их цель заключается в поиске работников на требуемые им вакансии. Веб–приложение по поиску вакансий может предоставить исполнение их целей;
* рекламодатели могут заинтересоваться веб–приложением по поиску вакансий, ведь они могут разместить свои собственные вакансии, находя активных пользователей и привлекая их к себе как потенциальных клиентов;
* партнеры могут являться заинтересованной стороной, которые могут предоставлять специализированную экспертную помощь или исполнять отведённую им роль в соответствии с требованиями пользователей;
* конкурентов может привлекать веб–приложения для получения ясности работы веб–приложения и могут использовать полученную информацию для собственных целей;
* регулирующие органы так же являются заинтересованной стороны, так как могут заинтересоваться вопросом безопасности конфиденциальных данных пользователей и соответствии веб–приложения законодательству;
* разработчики тоже могут заинтересоваться, ведь их работа состоит в разработке сайтов.

## 5.4 Устав проекта

Устав проекта – это документ, в котором излагается видение проекта, его масштаб, цели, основные заинтересованные стороны, а также то, как они будут осуществляться или реализовываться. Он также известен как заявление о проекте и отчет об определении проекта

Таблица 20 — Устав проекта

|  |  |
| --- | --- |
| Назначение проекта | Разработка веб–приложени по поиску вакансий |
| Цели проекта | Создать веб–приложение по поиску вакансий простым, доступным, многогранным и объёмным в своих возможностях |
| Описание проекта | Веб–приложение, которое предоставляет возможность соискателям найти подходящую им вакансию, а работодателям разместить своё объявление в поисках работников на требуемою им вакансию |
| Задачи проекта | 1. Провести анализ предметной области;  2. Определить проблемы схожих веб–приложений и найти варианты решении этих проблем  3. Разработать базу данных для хранения информации  4. Разработка интерфейса веб–приложения |
| Результаты проекта | Создание работоспособного веб–приложения по поиску вакансий |
| Критерии успешности выполнения проекта | Возможность создания резюме, оставление откликов и лайков на объявления для соискателей  Возможность работодателя добавлять объявления, редактировать из, менять статус и удалять. Так же осуществить функцию для просмотра резюме соискателя и скачать резюме в pdf формате |

## 5.5 Планирование проекта (WBS, диаграмма Ганта)

WBS – что переводиться как иерархическая структура работ, представляет из себя разбиение проекта на конкретные результаты, которые должны быть достигнуты для достижения целей проекта.

WBS один из самых эффективных способов для способа наглядно отразить весь объём проекта. Он фокусирует внимание не на процессе, а на ожидаемом результате, и создаёт нужный «посыл». С его помощью можно предотвратить риски и изменения, снизить их вероятность и влияние, так как именно здесь всплывут многие неочевидные ранее вещи.

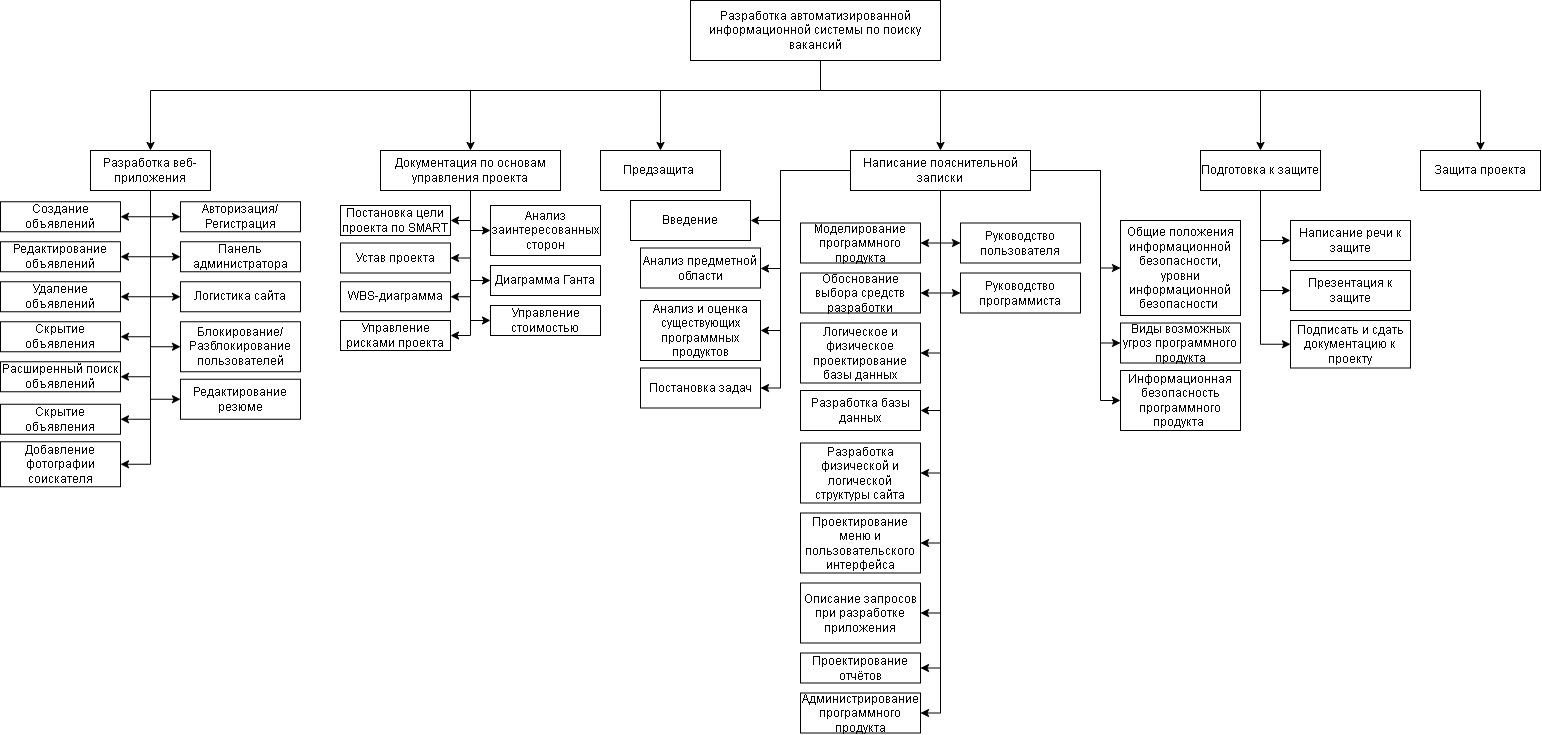


Рисунок 29 – WBS проекта

Планирование проекта – это непрерывный процесс определения наилучшего способа действий для достижения поставленных целей проекта с учётом складывающейся обстановки.

Процесс планирования осуществляется на протяжении всего жизненного цикла проекта, начиная с предварительного укрупненного плана в составе концепции проекта и оканчиваются детальным планом работ завершающей фазы проекта. При это происходит уточнение и детализация планов по мере прогресса проекта. На стадии планирования определяется организация, метода и средства управления осуществлением проекта, как целостной системы, так и в разрезе отдельных ее этапов и элементов.

Основным результатом стадии планирования является свободный план осуществления проекта, объединяющий результаты планирования по всем функциям управления проектом.

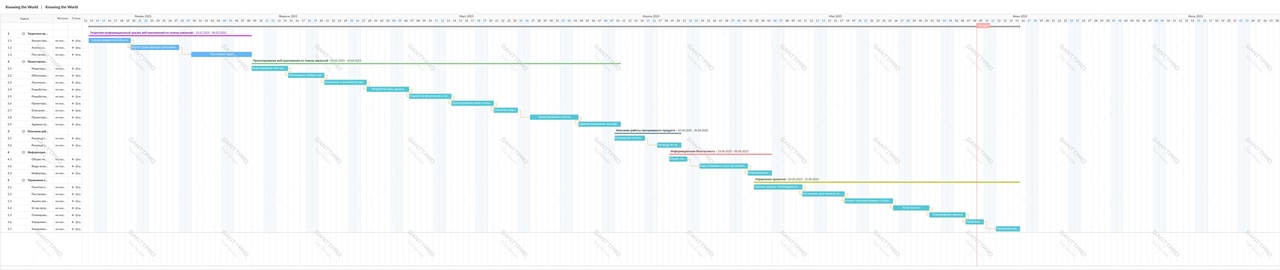


Рисунок 30 – Диаграмма Ганта

## 5.6 Управление стоимостью

**Управление стоимостью проекта** объединяет процессы, выполняемые в ходе планирования, разработки бюджета и контролирования затрат, обеспечивающие завершение проекта в рамках утвержденного бюджета.

Важной особенностью процессов управления стоимостью проекта является их очень тесная связь с иными процессами планирования. Информация о рисках проекта также может значительно повлиять как на величину, так и на структуру проекта.

Одним из первых шагов, выполняемых менеджером проекта в ходе управления стоимостью проекта, является разработка концепции управления стоимостью. Эта концепция должна содержать общие правила организации управления затратами проекта, принципы учета и документирования, рекомендуемые методики и технологии.

Разработка концепции управления стоимостью и финансированием проекта:

* выработка стратегии управления стоимостью и финансами проекта (определение целей и задач, критериев успеха и неудач, ограничений и допущений);
* проведение экономического анализа и обоснования проекта (маркетинг, оценка стоимости и источников финансирования, прогноз выполнения);
* общая экономическая оценка проекта;
* разработка укрупненного графика финансирования;
* определение требований к системе управления стоимостью и финансированием в проекте;
* утверждение концепции.

Тарифная ставка – фиксированный размер оплаты труда работника за выполнение нормы труда определенной сложности (квалификации) за единицу времени без учета компенсационных, стимулирующих и социальных выплат.

Тст = МРОТ / Вф

где МРОТ – минимальный размер оплаты труда (на данный момент размер минимальной заработной платы в Челябинской области составляет 16242 руб.);

Вф – отработанное время (168 час. в месяц, т.е. 21 смена \* 8 часов)

Тст = 16242 / 168 = 96,7 руб.

При создании проекта время работы компьютера:

Фв = 1,15 \* (Тп + Тд + Тотл) \* kкор,

Фв = 1,15 \* (78,125 + 19,68 + 26) \* 0,9 = 128 час.

**Себестоимость** — это затраты на производство и продажу продукции, проведение работ, предоставление услуг.

ЗПо = (26+ 78,125) \* 0,9 \* 96,7+ (11,25 + 78,125) \* 0,9 \* 96,7 +

+ (78,125 + 31,8 + 35) \*0,9 \* 96,7 = 29453руб.

ЗПд = 0,2 \* 29453 = 5890,6 руб.

ЕСН = (29453+5890,6)\*0,3 = 10603 руб.

2 % от стоимости оборудования = 1000 руб.

Зэ = 0,65 кВт\ч \* 3,47 руб. \* 128ч. = 288,7 руб.

С = 29453+ 5890,6+ 10603+ 1000 + 288,7 = 47235руб

Прибыль – это разница между общим доходом и себестоимостью товаров или услуг, которые были реализованы компанией за определенный период времени.

Пр = Вр – Зи,

где Вр – выручка от реализации программного продукта (деньги от заказчика);

Зи – итоговая сумма затрат на разработку

Пр = 71 363 – 47235= 24128 руб.

Таким образом, управление стоимостью дипломного проекта веб–приложения по поиску вакансий по МРОТ в Челябинской области составляет 24128 рублей.

## 5.7 Управление рисками проекта

Управление рисками проекта – это процесс поиска, оценки и предотвращения потенциальных проблем. Управление рисками не только снижает влияние негативных ситуаций на проект, оно так же высвобождает ресурсы.

Есть разные классификации рисков. Чаще всего упоминаются следующие риски:

* временные риски. Это вероятность того, что на выполнение задач в проекте уйдёт больше времени, чем запланировано;
* бюджетные риски. Из–за плохого планирования стоимости проекта может оказаться больше, чем заложено в бюджете;
* риски изменения объёмов работы. Они могут появиться, если исполнители не поняли требований к проекту или если были внесены кардинальные изменения;
* внешние риски. Это потенциальные события, которые находятся за пределами возможностей контролирования.

Таблица 21 — Карта риска проекта

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование риска | Условия риска | Последствие риска | Борьба с риском |
| Кража информации из базы данных | Взлом базы данных | Потеря важных данных | Разграничение доступа |
| Взлом административной панели | Взлом административной панели через перебор паролей | Кража данных пользователей | SSL, защита сервера (обновления, закрытие портов) смена периодическая – 1р в 2 месяца |

Продолжение таблицы 21 — Карта риска проекта

|  |  |  |  |
| --- | --- | --- | --- |
| Большая нагрузка на сайт | Нагрузка на сайт большим количеством пользователей или запросов | Медленная работа сайта | периодически проводить оптимизацию кода и загрузки сайта |
| Изменение структуры проекта | Резкая смена в структуре проекта, её целей, задач или функционала | Остаются брошенные и незавершённые задачи, полная или частичная переделка проекта | Поставить четкие цели и задачи ещё в начала проекта. Или же, принося изменение, сначала проанализировать, как это отразиться на текущем состоянии проекта |
| Плохая документация проекта | Неграмотная, хаотичная и не структурированная информация в документации | Потеря важной и необходимой информации, отсутствие контрольных показателей | Написание основательной проектировкой документации |
| Невыполнение проекта в срок | Неправильная планировка времени и ресурсов | Отсутствие полностью готово проекта, большое количество недочётов и багов | Грамотное планирование времени и распределение ресурсов при создании проекта |

# **ЗАКЛЮЧЕНИЕ**

В результате выполнения дипломного проекта было разработка автоматизированное веб–приложение по поиску вакансий на фреймфорке Laravel, с созданной базой данных в MySQL, выводом графиков через JavaScript и создание дизайна через фреймфорк Bootstrap. Весь предусматриваемый функционал для пользователей был полностью реализован. Разграничение ролей для пользователей было выполнено. Административная панель полностью функционирует и выдаёт статистку работы сайта.

В процессе разработки проекта были решены различные технические задачи, связанные с созданием функциональных модулей и осуществлением их взаимодействия с базой данных и другими частями приложения.

Были проведены работы с выходными документами в формате pdf. Файлы доступны для скачивания.

В результате выполнения дипломного проекта была проанализированная информация о конкурентных сайтах, выявлены их проблемы и недостатки. Основываясь на данной информации полученной в результате, было разработано веб–приложения, в котором были учтены все ранее обнаруженные проблемы сайтов для поиска работы.

# **СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ**

1. **Агальцов,** В.П. Базы данных: в 2 книгах. Книга 2. Распределенные и удаленные базы данных: учебник / В.П. Агальцов. – Москва: ФОРУМ: ИНФРА-М, 2021. – 271 с. – (Высшее образование: Бакалавриат). – ISBN 978-5-8199-0713-9. - Текст: электронный. – URL: https://znanium.com/catalog/product/1514118. – Режим доступа: по подписке.
2. **Ананьева, Т. Н.** Стандартизация, сертификация и управление качеством программного обеспечения : учебное пособие / Т.Н. Ананьева, Н.Г. Новикова, Г.Н. Исаев. — Москва : ИНФРА-М, 2021. — 232 с. — (Среднее профессиональное образование). - ISBN 978-5-16-014887-8. - Текст : электронный. - URL: https://znanium.com/catalog/product/1413308 (дата обращения: 03.05.2023). – Режим доступа: по подписке.
3. **Коваленко, В. В.** Проектирование информационных систем : учебное пособие / В.В. Коваленко. — 2-е изд., перераб. и доп. — Москва: ИНФРА-М, 2023. — 357 с. — (Высшее образование: Бакалавриат). — DOI 10.12737/987869. - ISBN 978-5-00091-783-1. - Текст : электронный. - URL: https://znanium.com/catalog/product/1894610 (дата обращения: 04.05.2023). – Режим доступа: по подписке.
4. **Кроксен-Джон Д.** Оптимизация интернет-магазина: Почему 95% посетителей вашего сайта ничего не покупают и как это исправить [Электронный ресурс] : Справочное пособие / Кроксен-Джон Д., Ван Тондер Й. - Москва :Альпина Паблишер, 2018. - 318 с.: ISBN 978-5-9614-7131-1 - Режим доступа: <https://znanium.com/read?id=333470>  
   (дата обращения: 20.01.2023)
5. **Курсы по программированию**: [Электронный ресурс] // URL: <https://htmlacademy.ru/> (дата обращения: 28.05.2023)
6. **Лаврентьев, А. Н.** Цифровые технологии в дизайне. История, теория, практика [Электронный ресурс] : учебник и практикум для вузов / А. Н. Лаврентьев [и др.] ; под редакцией А. Н. Лаврентьева. - 2-е изд., испр. и доп. - Москва : Издательство Юрайт, 2019. — 208 с. — (Авторский учебник). — ISBN 978-5-534-07962-3. — Режим доступа: <https://www.urait.ru/bcode/424029> (дата обращения: 06.04.2023)
7. **Лисьев, Г. А.** Программное обеспечение компьютерных сетей и web-серверов : учебное пособие / Г.А. Лисьев, П.Ю. Романов, Ю.И. Аскерко. — Москва : ИНФРА-М, 2023. — 145 с. — (Среднее профессиональное образование). - ISBN 978-5-16-014514-3. - Текст : электронный. - URL: https://znanium.com/catalog/product/1878635 (дата обращения: 01.04.2023). – Режим доступа: по подписке.
8. **Назаров, С. В.** Архитектура и проектирование программных систем : монография / С.В. Назаров. — 2-е изд., перераб. и доп. — Москва : ИНФРА-М, 2023. — 374 с. — (Научная мысль). — DOI 10.12737/18292. - ISBN 978-5-16-011753-9. - Текст : электронный. - URL: https://znanium.com/catalog/product/1895672 (дата обращения: 15.05.2023). – Режим доступа: по подписке.
9. **Немцова, Т. И.** Компьютерная графика и web-дизайн [Электронный ресурс] : учебное пособие / Т. И. Немцова, Т. В. Казанкова, А. В. Шнякин ; под ред. Л.Г. Гагариной. — Москва : ИД «ФОРУМ» : ИНФРА-М, 2019. — 400 с. + Доп. материалы. - Режим доступа: <https://znanium.com/read?id=329728> (дата обращения: 10.05.2023)
10. **Побединский, Е. В.** Проектирование веб-сайтов с использованием технологий PHP, HTML, CSS и WordPress : учебное пособие / Е. В. Побединский, В. В. Побединский. — Екатеринбург : УГЛТУ, 2018. — 115 с. — ISBN 978-5-94984-651-3. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/142518 (дата обращения: 04.05.2023). — Режим доступа: для авториз. пользователей.
11. **Портал по php, MySQL и другим веб-технологиям** [Электронный ресурс] ‑ Режим доступа : <http://www.php.su> (дата обращения: 15.04.2023)
12. **Ресурсы по программированию на php**: [Электронный ресурс] // URL: <http://www.php.net> (дата обращения: 29.04.2023)
13. **Ресурсы по программированию**: [Электронный ресурс] // URL: <http://www.realcoding.net/> (дата обращения: 01.05.2023)
14. **Шаньгин, В. Ф.** Информационная безопасность компьютерных систем и сетей [Электронный ресурс] : учебное пособие / В. Ф. Шаньгин. — Москва : ИД «ФОРУМ» : ИНФРА-М, 2019. — 416 с. — (Среднее профессиональное образование). - Режим доступа: <https://znanium.com/read?id=336332> (дата обращения: 10.04.2023)
15. **Шаньгин, В. Ф.** Комплексная защита информации в корпоративных системах [Электронный ресурс] : учебное пособие / В.Ф. Шаньгин. — Москва : ИД «ФОРУМ» : ИНФРА-М, 2019. — 592 с. — (Высшее образование: Бакалавриат). - Режим доступа: <https://znanium.com/read?id=330966> (дата обращения: 11.02.2023)
16. **Арлоу Джим**. UML 2 и унифицированный процесс. Практический объектно-ориентированный анализ и проектирование [Электронный ресурс] : учебное пособие // URL https://financial.lnu.edu.ua/wp-content/uploads/2020/09/Arlou\_i\_Neyshtadt\_UML\_i\_UP.pdf

# **ПРИЛОЖЕНИЕ А**

**(обязательное)**

**Листинг программы**

Листинг 1 – Фрагмент кода регистрации

registration.blade.php

@extends('layout')

@section('title')Регистрация@endsection

@section('main\_content')

<h2 class="text—info">Заполните все поля</h2>

<center><font color='red'>{{$errors}}</font></center>

@if ($reg==0)

<form action="/reg\_complete" method="POST">

@csrf

@method('post')

<div class="col">

<div class="card mb—4">

<div class="card—body text—dark">

<div class="col—sm—6">

<label for="firstName" class="form—label">Введите логин</label>

<input type="text" name="login" class="form—control" value="{{$arrayLogic['login']}}">

</div>

<div class="col—sm—6">

<label for="firstName" class="form—label">Введите пароль</label>

<input type="password" name="password" class="form—control" value="{{$arrayLogic['password']}}">

</div>

<div class="col—sm—6">

<label for="firstName" class="form—label">Подтвердите пароль</label>

<input type="password" name="password\_1" class="form—control" value="{{$arrayLogic['password\_1']}}">

</div>

<div class="col—sm—6">

<label for="firstName" class="form—label">Введите название огранизации</label>

<input type="text" name="organization\_title" class="form—control" value="{{$arrayLogic['organization\_title']}}">

</div>

<div class="col—sm—6">

<label for="firstName" class="form—label">Введите описание вашей организации</label>

<input type="text" name="organization\_description" class="form—control" value="{{$arrayLogic['organization\_description']}}">

</div>

<div class="col—sm—6">

<label for="firstName" class="form—label">Введите номер телефона</label>

<input type="text" maxlength="11" name="organization\_phone" class="form—control" value="{{$arrayLogic['organization\_phone']}}">

</div>

<div class="col—sm—6">

<label for="firstName" class="form—label">Введите почту</label>

<input type="text" name="organization\_email" class="form—control" value="{{$arrayLogic['organization\_email']}}">

</div>

<div class="col—sm—6">

<label for="firstName" class="form—label">Введите адрес</label>

<input type="text" name="organization\_adress" class="form—control" value="{{$arrayLogic['organization\_adress']}}">

</div>

<input type="hidden" name='type' value="0">

<pre>

<p><input type="submit" class="btn btn—primary btn—lg px—4 gap—3"></p>

</form>

</div>

</div>

</div>

@else

<form action="/reg\_complete" method="POST">

@csrf

@method('post')

<div class="col">

<div class="card mb—4">

<div class="card—body text—dark">

<div class="col—sm—6">

<label for="firstName" class="form—label">Введите логин</label>

<input type="text" name="login" class="form—control" value="{{$arrayLogic['login']}}">

</div>

<div class="col—sm—6">

<label for="firstName" class="form—label">Введите пароль</label>

<input type="password" name="password" class="form—control" value="{{$arrayLogic['password']}}">

</div>

<div class="col—sm—6">

<label for="firstName" class="form—label">Подтвердите пароль</label>

<input type="password" name="password\_1" class="form—control" value="{{$arrayLogic['password\_1']}}">

</div>

<div class="col—sm—6">

<label for="firstName" class="form—label">Введите Фамилию</label>

<input type="text" name="summary\_surname" class="form—control" value="{{$arrayLogic['summary\_surname']}}">

</div>

<div class="col—sm—6">

<label for="firstName" class="form—label">Введите Имя</label>

<input type="text" name="summary\_name" class="form—control" value="{{$arrayLogic['summary\_name']}}">

</div>

<div class="col—sm—6">

<label for="firstName" class="form—label">Введите Отчество</label>

<input type="text" maxlength="11" class="form—control" name="summary\_patronymic" value="{{$arrayLogic['summary\_patronymic']}}">

</div>

<div class="col—sm—6">

<label for="firstName" class="form—label">Введите свой возраст</label>

<input type="number" name="summary\_age" class="form—control" value="{{$arrayLogic['summary\_age']}}">

</div>

<div class="col—sm—6">

<label for="firstName" class="form—label">Введите свой город</label>

<input type="text" name="summary\_city" class="form—control" value="{{$arrayLogic['summary\_city']}}">

</div>

<div class="col—sm—6">

<label for="firstName" class="form—label">Введите свой номер телефона</label>

<input type="text" name="summary\_phone" class="form—control" maxlength='11' value="{{$arrayLogic['summary\_phone']}}">

</div>

<div class="col—sm—6">

<label for="firstName" class="form—label">Введите свой email</label>

<input type="text" name="summary\_email" class="form—control" maxlength='64' value="{{$arrayLogic['summary\_email']}}">

</div>

<div class="col—sm—6">

<label for="firstName" class="form—label">Ваш опыт</label>

<select name="summary\_experience" class="form—select">

@foreach($experience as $exp)

<option value="{{$exp—>id}}">{{$exp—>experience\_name}}</option>

@endforeach

</select>

</div>

<div class="col—sm—6">

<label for="firstName" class="form—label">Желаемая должность</label>

<select name="summary\_post" class="form—select">

@foreach($post as $p)

<option value="{{$p—>id}}">{{$p—>post\_name}}</option>

@endforeach

</select>

</div>

<div class="col—sm—6">

<label for="firstName" class="form—label">Ваше образование</label>

<select name="summary\_education" class="form—select">

@foreach($education as $e)

<option value="{{$e—>id}}">{{$e—>education\_name}}</option>

@endforeach

</select>

</div>

<div class="col—sm—6">

<label for="firstName" class="form—label">Выберите языки</label>

<select name="summary\_language" class="form—select">

@foreach($language as $l)

<option value="{{$l—>id}}">{{$l—>language\_name}}</option>

@endforeach

</select>

</div>

<div class="col—sm—6">

<label for="firstName" class="form—label">Ваша специализация</label>

<select name="summary\_specialization" class="form—select">

@foreach($specialization as $spec)

<option value="{{$spec—>id}}">{{$spec—>specialization\_name}}</option>

@endforeach

</select>

</div>

<div class="col—sm—6">

<label for="firstName" class="form—label">Ваш опыт вождения</label>

<select name="summary\_driving" class="form—select">

@foreach($driving as $d)

<option value="{{$d—>id}}">{{$d—>driving\_name}}</option>

@endforeach

</select>

</div>

<div class="col—sm—6">

<label for="firstName" class="form—label">Выберите вид занятости</label>

<select name="summary\_employment" class="form—select">

@foreach($employment as $e)

<option value="{{$e—>id}}">{{$e—>employment\_name}}</option>

@endforeach

</select>

</div>

<div class="col—sm—6">

<label for="firstName" class="form—label">Выберите график</label>

<select name="summary\_schedule" class="form—select">

@foreach($schedule as $sc)

<option value="{{$sc—>id}}">{{$sc—>schedule\_name}}</option>

@endforeach

</select>

</div>

<input type="hidden" name='type' value="1">

<br>

<p><input type="submit" class="btn btn—primary btn—lg gap—3"></p>

</div>

</div>

</div>

</form>

@endif

@endsection