Некоммерческое акционерное общество
«АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ»
Кафедра IT-инжиниринг

ДОПУЩЕН К ЗАЩИТЕ
Заведующий кафедрой
PhD, доцент
_____ Т.С. Картбаев
«____» _____ 2018 г.

ДИПЛОМНЫЙ ПРОЕКТ

На тему: Технологии разработки web-приложений (на примере компании «Web Line»).

Специальность 5В070400 – «Вычислительная техника и программное обеспечение»

Выполнил Киркица А. М. Группа ВТ-14-2
Научный руководитель к.ф.-м.н., доцент Турганбаев Е. С.

Консультанты:
по экономической части: к.э.н., профессор _____ Ж.Г. Аренбаева
«23» июня 2018 г.

по безопасности жизнедеятельности: к.т.н, доцент _____ А.А. Абикенова
«23» 05 2018 г.

по применению
вычислительной техники: ст. преп. _____ А.М. Рамазанова
«22» 05 2018 г.

Нормоконтролер: PhD, ст. преп. _____ Ж. Бидахмет
«31» 05 2018 г.

Рецензент: PhD, асс. проф. _____ О. Б. Баймуратов
«____» _____ 2018 г.

Алматы 2018

Некоммерческое акционерное общество
«АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ»

Институт систем управления и информационных технологий

Кафедра IT-инжиниринг

Специальность 5В070400 – «Вычислительная техника и программное обеспечение»

ЗАДАНИЕ
на выполнение дипломного проекта

Студенту Киркица Александру Максимовичу

Тема проекта: Технологии разработки web-приложений (на примере компании «Web Line»)

Утверждена приказом по университету № 155 от «23» октября 2017 г.
Срок сдачи законченной работы « 1 » июня 2018 г.

Исходные данные к проекту (требуемые параметры результатов исследования (проектирования) и исходные данные объекта): HTML, CSS, JQuery – веб-интерфейс, Joomla – система управления сайтом, PostgreSQL – система управления базами данных, Java - язык программирования, Apache Tomcat – контейнер сервлетов.

Перечень вопросов, подлежащих разработке в дипломном проекте, или краткое содержание дипломного проекта:

а) проектирование базы данных с возможностью расширения;
б) функциональные возможности PostgreSQL;
в) вопросы безопасности жизнедеятельности и охраны труда;
г) экономическая эффективность работ по стандартизации.

Перечень графического материала (с точным указанием обязательных чертежей): представлены 5 таблиц, 34 иллюстрации.

Основная рекомендуемая литература:

1 Бакор А. Apache Tomcat для профессионалов – СПб., 2005 – 544 с.
2 Гербер Шилдт Java 8. Полное руководство 9-е издание 2015 – 1344 с.

3 Распределенная обработка данных: курс лекций / Сост. Найханова Л.В. – Улан-Удэ, Издательство ВСГТУ, 2001. – 122 с.

Консультации по работе с указанием относящихся к ним разделов проекта

Раздел	Консультант	Сроки	Подпись
Экономическая часть	Аренбаева Ж. Г	2.03.18-23.05.18	
Безопасности жизнедеятельности	Абикенова А.А.	14.03 – 23.05.18	
Программная часть	Рамазанова А.М.	05.05-23.05.18	
Нормоконтролер	Бидахмет Ж.	20.05.18 - 31.05.18	Ж. Бидахмет

График
подготовки дипломного проекта

Наименование разделов, перечень разрабатываемых вопросов	Сроки представления научному руководителю	Примечание
Исследование и описание предметной области	5.02.2018 – 19.02.2018	
Проектирование информационной модели	19.02.2018 – 12.03.2018	
Проектирование и создание базы данных	12.03.2018 – 02.04.2018	
Разработка веб-интерфейса портала и серверной части	02.04.2018 – 25.04.2018	
Безопасность жизнедеятельности	14.03.2018 – 23.05.2018	
Технико – экономическое обоснование	02.03.2018 - 23.05.2018	

Дата выдачи задания « 23 » октября 2017 г.

Заведующий кафедрой _____ Т.С. Картбаев

Научный руководитель работы _____ Е. С. Турганбаев

Задание принял к исполнению студент _____ А. М. Киркица

Аңдатпа

Осы дипломдық жобанын максаты, қағаз бетіндегі адам немесе ұйым жайлы ақпаратты электронды нұскаға видео, аудио, графикалық контент, интерактивті карта жане тағыда басқа арқылы веб-портал құру болып табылады. Дипломдық жоба веб-порталдың барлық функционалы мен интерфейсін сипаттайды. Мобильдік қосымшалармен жалпы ақпараттық жүйені құруға мүмкіндік беретін, сайттарды дамытудың заманауи технологиялары қарастырылады Ақпараттық жүйенің дерекқор құрылымы әзірленді.

Веб-портал келесі технологияларды қолдана отырып әзірленеді: сервер бөлігі JAVA технологиясы және объектілі-реляциялық деректер базасын басқару жүйесін қолданатын PHP арқылы PostgreSQL базасын басқару. Веб-интерфейсті дамыту үшін CMS Joomla және Bootstrap веб-негіздері пайдаланылды.

Аннотация

Целью данного дипломного проекта является создание веб-портала, предоставляющего замену бумажному носителю контактной информации о человеке или организации на электронную версию, при этом расширив представительские возможности визитки с помощью видео, аудио, графического контента, интерактивных карт и пр. В дипломном проекте описывается весь функционал и интерфейс веб-портала. Рассмотрены современные технологии разработки сайтов, позволяющие создавать общую информационную систему с мобильными приложениями. Спроектирована структура базы данных информационной системы.

Веб-портал разработан с применением следующих технологий: серверная часть реализована на технологии JAVA сервлетов и PHP с использованием свободной объектно-реляционной система управления базами данных PostgreSQL. Для разработки веб-интерфейса использована CMS Joomla и веб-фреймворк Bootstrap.

Annotation

The purpose of this diploma project is creation of a web portal, that provides a replacement for a paper card with contact information about a person or an organization to an electronic version, while expanding the representative possibilities of a business card with video, audio, graphic content, interactive maps, etc. The diploma project describes all the functionality and a web portal interface. The modern technologies of development of sites, that allow creation of the general information system with mobile applications are considered. The database structure of the information system has been designed.

The web portal is developed using the following technologies: the server part is implemented on the JAVA servlets technology and PHP using the free object-relational database management system PostgreSQL. CMS Joomla and the Bootstrap web framework were used for the development of the web interface.

Содержание

Введение

Сегодня информационные технологии плотно вошли в нашу жизнь, во все сферы её деятельности. Большая часть информации была перенесена с бумажных носителей на цифровые. Идеей моего дипломного проекта является оцифровка визитной карточки. Визитная карточка – это карточка, на которой указана основная контактная информация человека. Стандартный размер визиток в СНГ – 90мм х 50мм. Бумажные визитки очень ограничены в количестве информации, которую они могут передать. Для того, чтобы создать визитку, необходимо большое количество времени на разработку дизайна и поход в типографию. Электронную же визитку можно создать за 5 минут: для этого нужно просто зайти на сайт и зарегистрироваться. Большим преимуществом электронных визитных карточек является то, что они не заканчиваются, и если у вас меняется какая-то контактная информация, то вы просто редактируете её в своем профиле и она автоматически меняется у всех обладателей вашей визитки.

Целью данного дипломного проекта является создание веб-портала (а в дальнейшем и приложения), предоставляющего замену бумажному носителю контактной информации о человеке или организации на электронную версию, при этом расширив представительские возможности визитки с помощью видео, аудио, графического контента, интерактивных карт и пр.

В данном дипломном проекте будут рассмотрены описание внутренней архитектуры системы, сравнение и описание технологий, использованных при ее создании, проведен анализ средств, показан конечный интерфейс веб-приложения.

Дипломный проект содержит реализацию клиентской стороны пользовательского интерфейса и программно-аппаратной части сервиса.

Используемые технологии: серверная часть реализована на технологии JAVA сервлетов и PHP с использованием свободной объектно-реляционной система управления базами данных PostgreSQL. Для разработки веб-интерфейса использована CMS Joomla и веб-фреймворк Bootstrap.

1 Проектирование структуры веб-портала

1.1 Основная целевая аудитория веб-портала

Определение целевой аудитории – важный пункт при создании любого веб-приложения. Чем точнее мы сможем определить и описать того, кто будет пользоваться создаваемым веб-порталом, тем удобнее и практичнее получится программный продукт. Для данного веб-портала была определена следующая целевая аудитория:

1) Физические или юридические лица, занимающиеся распространением товаров и услуг и нуждающиеся в распространении и рекламировании информации о своей профессиональной деятельности среди существующих и потенциальных клиентов.

2) Физические или юридические лица, нуждающиеся в поиске информации и рекомендаций о товарах и услугах.

1.2 Разделы меню веб-портала

На веб-портале реализованы следующие разделы (страницы):

1) «Регистрация» с пошаговым вводом информации и подтверждением номера телефона через СМС. Используя регистрацию пользователей с подтверждением номера телефона через СМС значительно уменьшается вероятность создания спам-аккаунтов. В случае, если пользователь забыл свой пароль, он также сможет восстановить доступ к своему аккаунту отправив на него СМС с кодом.

2) «Каталог визиток» – список визиток, который находится в общем доступе, каталог оснащен фильтрами, поиском и сортировками, а также внутренней вкладкой настроек.

3) «Визитница» – список визиток, состоящий из чужих визиток, которые пользователь сохранил у себя в аккаунте, список имеет фильтры, поиск и сортировку, а также внутреннюю вкладку настроек.

4) «Мои Визитки» - личные визитки пользователя, доступно создание трёх визиток: публичная, приватная и фирменная. Публичная визитка доступна для просмотра абсолютно всем пользователям сервиса. Доступ к приватным визиткам имеют только те пользователи, которым дал доступ владелец визитки. Фирменная визитка регистрируется на юридическое лицо и представляет собой визитку организации.

5) «Сообщения».

6) «Встречи» – созданные в системе записи какими-либо пользователями о физической встрече с другими пользователями.

7) «Настройки».
8) «Вход» / «Выход».

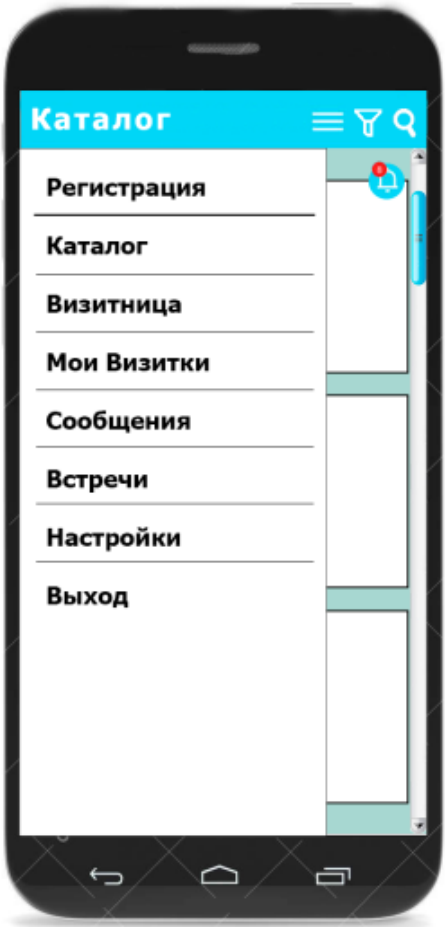


Рисунок 1.1 – Меню веб-портала, адаптированное под мобильное устройство

1.3 Регистрация в системе через веб-портал

После запуска регистрации пользователю пошагово будут открываться следующие всплывающие окна:

Шаг 1: Ввод номера телефона.

После ввода номера пользователем проверяется наличие данного номера в системе, если номер существует, то происходит отказ от регистрации и выводится надпись: «Данный номер уже зарегистрирован в системе, проверьте введенный номер или введите другой».

Если номер проходит проверку, то выводится сообщение: «Продолжить регистрацию».

На данном этапе пользователь может по указанной ссылке открыть окно с Договором оферты и ознакомиться с ним, после прочтения пользователь возвращается обратно на данную вкладку.

Возле кнопок «Продолжить» и «Отмена» располагается надпись: «Продолжая регистрацию, вы подтверждаете, что ознакомились с договором оферты и полностью согласны с ним».

Нажимая кнопку «Отмена», регистрация полностью прекращается. Нажимая кнопку «Продолжить» открывается второй шаг.

Шаг 2: Ввод кода активации.

Код активации отправляется посредством СМС-сообщения на указанный номер телефона.

Это позволит подтвердить, что указанный номер телефона существует, и он принадлежит регистрируемому пользователю.

Если код по какой-либо причине не подходит, то можно запросить код повторно по кнопке «Получить код повторно».

После этого в системе генерируется UUID, который является основным идентификатором аккаунта, который не может быть изменен.

Шаг 3: Создание пароля.

Для получения доступа ко всем функциям пользователь должен придумать и ввести пароль. Пароль необходим для авторизации.

Восстановление пароля может происходить через телефон, email или через механизм подтверждения другом. При восстановлении пароля через друга, любому доверенному контакту, у кого есть аккаунт AdveCard (AdveCard – рабочее название веб-портала) посылается сообщение подтвердить личность того, кто восстанавливает пароль и при подтверждении, пароль будет скинут и откроется окно восстановления пароля

На данном шаге регистрация будет считаться завершенной и пользователю открывается страница «Моих визиток».

Все остальные данные собираются в необязательном порядке, и предлагаются в различные моменты работы с порталом.

Дополнительные сведения, которые будут собраны сайтом:

Язык: По умолчанию портал будет работать на русском языке, но при его разработке учитывается возможность дальнейшего перевода на другие языки.

По мере перевода локализации портала, языки будут добавляться, и пользователи смогут переключать их в настройках

Email: При открытии портала время от времени будет выводиться сообщение о том, что необходимо защитить аккаунт, указав свой email. При вводе email пользователем, на указанную почту высылается письмо, со ссылкой активации, при нажатии на которую происходит подтверждение. Таким образом, мы узнаем, что email действительно существует, и он принадлежит владельцу. В

последующем, при утере мобильного номера или при оспаривании аккаунта другими пользователями, восстановить права на аккаунт можно будет по email.

Прочие номера телефонов: После регистрации пользователь может добавлять к своему аккаунту другие номера телефонов. Любой телефон можно удалить из аккаунта, даже если это номер, по которому происходила регистрация.

Телефоны могут пройти процедуру подтверждения через код по СМС, в данном случае телефоны будут указаны как проверенные.

В любой момент времени в списке телефонов аккаунта должен быть хотя бы один подтвержденный номер.

Имя и фамилия: Ввод имени и фамилии необходим при создании Персональной визитки, и добавлятся к аккаунту после ее первого создания. При необходимости пользователь может изменить имя и фамилию в настройках.

Страна резидентства: Страна определяется по номеру телефона, в котором указан оператор, чья страна заведомо известна. При необходимости пользователь может изменить страну резидентства в настройках.

Населенный пункт: Данное значение необходимо для фильтрации публичных визиток, чтобы отображать только визитки, которые действуют на территории населенного пункта. Это значение можно менять для того, чтобы можно было искать визитки в других местах и также представить свою визитку на других территориях.

Населенный пункт определяется:

- по указанному при регистрации номеру телефона (устанавливается после регистрации по умолчанию);
- по географическим координатам, определенным по GPS;
- по мобильным данным;
- по IP;
- также пользователь может самостоятельно выбрать точку местоположения указанием точки на карте.

Пользователь может выбрать масштаб представления на трех уровнях, на уровне страны, области или населенного пункта. По указанным координатам система определит страну, область и населенный пункт. При этом если пользователь выбрал масштаб, например, населенный пункт, то его Публичные визитки будут «видны» только для пользователей, которые в фильтре Каталога выбрали также тот же населенный пункт, область или страну, в котором находится этот город.

1.4 Авторизация в системе

На веб-портале AdveCard можно работать анонимно или авторизовано:

Анонимная работа – никакой аутентификации не требуется. Такой режим устанавливается если пользователь еще не зарегистрировался в системе и/или не авторизовался.

Анонимно на портале можно просматривать только Каталог, производить в нем фильтрацию и сортировку визиток, а также запустить процедуру регистрации и/или провести аутентификацию

Авторизованная работа – для аутентификации необходимо ввести номер телефона и пароль. Если пользователь не сделал выход из системы, то после закрытия портала его сессия не пропадает, и при повторном открытии заново аутентифицироваться не нужно.

Авторизованным пользователям доступны все функции портала. Вход и выход открываются через главное меню.

Каталог визиток

Каталог визиток – это общая лента Публичных визиток, которая доступна всем пользователям, в том числе и анонимным.

После создания Публичной Визитки, пользователь может разместить ее в Каталоге, и теперь любой, кто откроет портал AdveCard может найти ее, просмотреть данные, разрешенные владельцем для публичного просмотра, рекомендовать ее другим пользователям и связаться с владельцем.

Рисунок 1.2 – Каталог визиток

Пользователь в любой момент может удалить свою Публичную Визитку из Каталога.

В Каталоге можно производить фильтрацию и сортировку, а также делать запросы.

Визитка в Каталоге будет представлять аналог физической визитки. При переходе на страницу визитки появится полное представление Визитки с полной персональной информацией, видео-блоком и другими медиа-данными. В видео-блоке можно просмотреть все существующие видео для данной визитки. Обратно попасть в Публичную Визитницу можно нажав на кнопку «Назад».

В шапке справа указывается раздел: «Каталог», слева 3 кнопки: меню, фильтр и поиск.

Ниже шапки справа, при условии, когда есть новые сообщения, появляется круглая кнопка с колокольчиком и количеством новых сообщений, по нажатию на который откроется окно с сообщениямия.

По нажатию кнопки меню справа показывается меню портала.

По нажатию кнопки фильтра справа показывается шторка с фильтрами.

Доступные фильтры:

- по региону (страна, область, город);
- по видам деятельности;
- по специальности;
- по компании.

По умолчанию Публичная Визитница отфильтрована по региону, установленному в значение из настройки аккаунта «Населенный пункт».

В фильтре по региону возможны следующие виды масштабирования территорий: в пределах страны, в пределах области (административно-территориальной единицы (АТЕ) страны) и в пределах населенного пункта.

Если пользователь выбрал масштаб «страна» и в поле «страна» выбрал, например, Казахстан, то в Каталоге останутся все Визитки, которые действуют на территории только Казахстана.

После выбора страны можно выбрать Административно-территориальную единицу.

Если пользователь выбрал масштаб АТЕ и в поле «АТЕ» выбрал, например, Алматинская область, то в Каталоге останутся все Визитки, которые действуют на территории только Алматинской области.

После выбора АТЕ можно выбрать населенный пункт.

Если пользователь выбрал масштаб АТЕ и в поле «Населенный пункт» выбрал, например, Алматы, то в Каталоге останутся все Визитки, которые действуют на территории только города Алматы.

Если система определит, что устройство пользователя находится на другой территории, отличной от настройки «Населенный пункт», то при открытии Каталога будет предложено переустановить «населенный пункт» на местный, после чего каталог отфильтруется по-новому региону.

Если при открытой сессии пользователь переставит фильтр вручную, то до закрытия веб-портала или нажатия кнопки «Вернуть значения по умолчанию» данный фильтр изменяться не будет.

По нажатию на кнопку поиск сверху появляется поле ввода текста поиска. Для поиска по произвольной строке можно вводить ключевые слова. Поиск производится по всем доступным значениям в визитке. Последовательность ключевых слов, разделенных пробелом не учитывается, каждое слово ищется отдельно. Результатом будет объединение всех визиток, имеющих хоть одно слово из заданных в поиске

В настройках можно устанавливать сортировку для порядка отображения Визиток в Каталоге. Доступные сортировки:

- по дате добавления;
- по имени;
- по фамилии;
- по рейтингу.

По умолчанию Каталог отсортирован по рейтингу и дате добавления.

На визитке в каталоге стоит кнопка «Я рекомендую», по нажатию на которую пользователь увеличивает рейтинг Визитки, свой голос можно удалить из рейтинга, повторно нажав на данную кнопку.

Визитница

Визитница – это список визиток, которые были добавлены самим владельцем визитницы, и доступна только авторизованным пользователям.

В Визитнице можно производить фильтрацию и сортировку, а также делать запросы.

Визитница выглядит идентично Каталогу, но с другим набором фильтров и сортировок.

Доступные фильтры:

- по региону (страна, область, город);
- по видам деятельности;
- по специальности.

Доступные сортировки:

- по дате.

По умолчанию Каталог отсортирован по рейтингу и дате фамилии.

Визитница не зависит от настройки «Населенный пункт», все Визитки в ней не скрываются в других регионах.

Визитки можно добавлять и удалять из Визитницы без ограничений.

Создание Визитки

Каждый пользователь может создать две Персональные и одну Фирменную Визитки, которые можно создать в разделе «Мои Визитки», выбором соответствующих кнопок.

Регистрация Персональной Визитки

Персональная Визитка – это набор персональных данных пользователя и информации по какой-либо его деятельности, оформленных по шаблону в единую визуальную форму.

Во время создания Персональной Визитки пользователю пошагово будут открываться следующие вкладки:

Шаг 1: Выбор языка для создания визитки

Данный язык будет применяться для создания и отображения данных на визитке. На каждую визитку, после завершения ее создания, можно добавить другие языки

Язык визитки не зависит от языка веб-портала.

Шаг 2: Выбор доступности визитки

Визитки имеют уровень доступности: приватные или публичные.

Приватные Визитки могут быть переданы другим пользователям только Владельцем. Рекомендация Приватной Визитки может проводиться только с согласия владельца. Приватные Визитки не отображаются в Публичных Визитницах.

Публичными Визитками могут делиться все со всеми, т.е. можно свободно рекомендовать и передавать чужие Публичные Визитки третьим лицам. Публичные Визитки можно выставлять в Публичной Визитнице.

Шаг 3: Ввод персональных данных:

- Фамилия*;
- Имя*;
- Отчество;
- Пол.

* - обязательные поля

Фамилию, имя и отчество, а также любую текстовую информацию далее нужно вводить на языке, выбранном в начале.

При создании первой Персональной Визитки фамилия и имя пользователя будут записаны в настройки портала.

Шаг 4: Ввод профессиональных данных:

- Отрасль деятельности* – выбирается из списка или, при отсутствии нужной, устанавливается галочка прочее и значение прописывается вручную. Отрасли деятельности должны отражать только общие представления без глубокой детализации.
- Специальность* – прописывается вручную.

- Место работы – выбирается из списка Фирменных Визиток, при отсутствии нужной, устанавливается галочка прочее и значение прописывается вручную.

- Должность – прописывается вручную.

- Описание своей деятельности.

- Режим работы – указанием дней недели и соответствующих времени начала и окончания работы в эти дни.

* - обязательные поля

Шаг 5: Контактные данные

- Номера телефонов* – если номеров несколько, то каждый номер вводится отдельно. К каждому номеру можно добавить комментарий, например, «факс» или «домашний» и т.п. По умолчанию в список номеров добавляются зарегистрированные в аккаунте номера, которые можно удалить из визитки.

- Email – если их несколько, то каждый вводится отдельно. К каждому email можно добавить комментарий. Каждый введенный email приложение будет проверять, отправляя на эти адреса код активации. Не подтвержденные email будут промаркированы как «Не подтвержденные».

- Ссылки на социальные сети: facebook, twitter, vk, linkedin, Instagram.

- Адреса* (указанием точки на карте) – адрес можно не отображать, но нужно выбрать для указания региона, который будет применяться в поиске. При создании первой Визитки в настройках приложения будет установлено значение «Населенный пункт».

* - обязательные поля

В Публичных Визитках при вводе контактов для каждого из них будет предложено выбрать уровень доступности контакта:

- Разрешить просматривать в Каталоге

- Разрешить просматривать только в Визитницах

После данного шага Персональная Визитка будет создана и отображена пользователю. После чего можно будет дополнить Визитку следующими элементами:

- Выбор фона – приложение предложит на выбор несколько вариантов фонов Визиток. По умолчанию будет установлен стандартный фон

- Видео презентации. В приложение возможно загрузить 3 вида видео презентаций:

1) Видео презентация личности – можно записать видео длительностью 10-30 секунд, в котором можно представиться и рассказать о себе;

2) Видео презентация бизнеса – можно записать видео длительностью 10-30 секунд, в котором нужно записать информацию о своей деятельности;

3) Видео презентация акции – можно записать видео длительностью 10-30 секунд с информацией о проводимой акции или скидках. На данное видео нужно указать текстовую информацию для поиска другими людьми

Для хранения видео используется видеохостинг – Vimeo. Каждая презентация может быть длиной только от 10 до 30 секунд.

Загружаемое видео должно соответствовать всем требованиям Vimeo.

Запись можно проводить с мобильного устройство непосредственно через портал AdveCard или загрузить заранее приготовленное видео.

- Добавление языков для визитки и ввод переводов информации.

Каждую визитку можно перевести на не ограниченное количество языков, для этого надо добавить язык к Визитке и сделать синхронный перевод всей текстовой информации. А также записать или загрузить видео на добавленном языке.

- Загрузка личной фотографии.
- Теги для поиска – можно указать теги для Визитки, по которой ее можно найти.
- Текстовые блоки – это любые тексты ограниченной длины, которые можно будет разместить на визитке. Например, это может быть слоган или рекламная информация.

Регистрация Фирменной Визитки.

Фирменная Визитка – это набор данных по организации, оформленных по шаблону в единую визуальную форму.

Шаг 2: Ввод наименования и выбор вида собственности.

Виды собственности нужно выбрать из предложенного списка, при отсутствии нужного, устанавливается галочка прочее и значение прописывается вручную.

Шаг 5: Выбор вида гос. идентификатора юридического лица и ввод самого гос. идентификатора

Государственные идентификаторы – это идентификаторы компаний в системе, без их указания, будет применяться номер телефона того, кто зарегистрировал Фирменную визитку

Шаг 6: Ввод отрасли деятельности.

Выбирается из списка или, при отсутствии нужной, устанавливается галочка прочее и значение прописывается вручную.

Шаг 7: Контактные данные.

- Номера телефонов – если номеров несколько, то каждый номер вводится отдельно. К каждому номеру можно добавить комментарий, например, «факс» или «ресепшен» и т.п.
- Email – если их несколько, то каждый вводится отдельно. К каждому email можно добавить комментарий.

- Ссылки на социальные сети и каналы компании: facebook, twitter, vk, linkedin, Instagram, youtube.
- Адреса (указанием точки на карте)* – указанная точка определит Страну резидентства и населенный пункт. Адрес будет предложен из сервисов Google.

* - обязательные поля

После данного шага Фирменная Визитка будет создана и отображена пользователю. После чего можно будет дополнить Визитку следующими элементами:

- Выбор фона – приложение предложит на выбор несколько вариантов фонов Визиток.
- Видео презентации. В приложении возможно загрузить 2 вида презентаций:

1) Видео презентации Компании – можно записать видео длительностью 10-30 секунд, в котором можно рассказать о Компании;
2) Видео презентации акции – можно записать видео длительностью 10-30 секунд с информацией о проводимой акции или скидках. На данное видео нужно указать текстовую информацию для поиска другими людьми;

Для хранения видео используется видеохостинг – Vimeo. Каждая презентация может быть длиной только от 10 до 30 секунд.

Загружаемое видео должно соответствовать всем требованиям Vimeo.

Запись можно проводить с мобильного устройства непосредственно через веб-портал AdveCard или загрузить заранее приготовленное видео.

- Добавление языков для визитки и ввод переводов информации.

Каждую визитку можно перевести на не ограниченное количество языков, для этого надо добавить язык к Визитке и сделать синхронный перевод всей текстовой информации. А также записать или загрузить видео на добавленном языке.

- Загрузка изображение логотипа.
- Теги для поиска – можно указать теги для Визитки, по которой ее можно найти.
- Текстовые блоки – это любые тексты ограниченной длины, которые можно будет разместить на визитке. Например, это может быть слоган или рекламная информация.

1.5 Персональная Визитка

Персональная визитка состоит из следующих блоков:

- Блок персональной информации;
- Блок Видео презентаций;
- Блок управления;

- В блоке персональной информации располагаются:
- ФИО*;
- Фотография;
- вид деятельности*;
- специальность*;
- описание деятельности;
- место работы (со ссылкой на Фирменную Визитку или без);
- должность;
- телефоны*;
- email;
- ссылки на соцсети: facebook, twitter, VK, linkedin, Instagram;
- описание деятельности.

* - обязательные поля

Рисунок 1.3 – Страница персональной визитки

Блок персональной информации, по виду напоминает физическую визитку, с текстовой информацией о специалисте. В зависимости от настроек, контактные данные владельца Визитки могут быть скрыты, для связи при этом можно воспользоваться специальной кнопкой «Попросить связаться», при нажатии на которую владельцу Визитки отправляется сообщение «[ФИО запрашивающего] с номером телефона [телефон запрашивающего] интересуется вашими контактами».

На визитке стоит кнопка «Я рекомендую», по нажатию на которую пользователь увеличивает рейтинг Визитки, свой голос можно удалить из рейтинга, повторно нажав на данную кнопку.

В блоке Видео презентаций располагается окно видео плеера, в котором можно просмотреть: видео-презентацию о владельце Визитки, или о его бизнесе или о проводимых им акциях.

Ниже окна видеоплеера располагаются кнопки, которые переключают видео-презентации.

Блок управления предоставляет клиенту возможность произвести действие указанной Визиткой:

- кнопка «Поделиться», по нажатии на которую открывается список способов отправки, где можно выбрать мессенджер или свои контакты, или список из своей Персональной Визитницы, где нужно выбрать один или несколько контактов, после чего рекомендованная Визитка будет отправлена адресатам;
- кнопка «Передать», по нажатию на которую будет открыто окно Встречи (обсудить);
- кнопка «Позвонить», по нажатию на которую будет отрыто приложение для звонка с основным номером Визитки;
- кнопка «Попросить связаться», по нажатии на которую можно послать сообщение владельцу Визитки, после чего владельцу Визитки будет послано сообщение, о том, что его Визиткой заинтересовались.

Редактирование визиток

Если пользователь редактирует визитку, то при изменении Персональных данных и данных по профессиональной деятельности, все клиенты, имеющие у себя в визитницах измененную Визитку, вместо контента Визитки увидят сообщение, что данные по Визитке были изменены. Клиенты смогут выбрать или сохранить измененную Визитку или удалить ее из своей Визитницы.

При изменении контактов, сообщения не выводятся, но контакты обновляются.

1.6 Фирменная Визитка

Фирменная Визитка состоит из следующих блоков:

- Блок информации о Компании;
- Блок Видео презентаций;
- Блок управления.

В блоке информации о Компании располагаются:

- наименование компании*;
- вид деятельности*;
- страна резидентства;
- адрес;
- телефоны компании*;
- корпоративный email;
- корпоративные ссылки на соцсети: facebook, twitter, VK, linkedin, Instagram;
- логотип;
- слоган.

* - *обязательные поля*

На визитке стоит кнопка «Я рекомендую», по нажатию на которую пользователь увеличивает рейтинг Визитки, свой голос можно удалить из рейтинга, повторно нажав на данную кнопку.

В блоке Видео презентаций располагается окно видео плеера, в котором можно просмотреть: видео-презентацию о Компании или о проводимых ею акциях.

Ниже окна видеоплеера располагаются кнопки, которые переключают видео-презентации.

Блок управления предоставляет клиенту возможность произвести действие указанной Визиткой:

- кнопка «Поделиться», по нажатии на которую открывается список способов отправки, где можно выбрать мессенджер или свои контакты, или список из своей Персональной Визитницы, где нужно выбрать один или несколько контактов, после чего рекомендованная Визитка будет отправлена адресатам
- кнопка «Передать», по нажатию на которую будет открыто окно Встречи (обсудить)
- кнопка «Позвонить», по нажатию на которую будет отрыто приложение для звонка с основным номером Визитки
- кнопка «Попросить связаться», по нажатии на которую можно послать сообщение владельцу Визитки, после чего владельцу Визитки будет послано сообщение, о том, что его Визиткой заинтересовались

Правообладатель компании может пожаловаться на Фирменную Визитку, созданную не им, и при доказательстве прав, закрепить Фирменную Визитку к своему аккаунту и отредактировать ее.

1.7 Рекомендации Визиток

Для распространения Визиток можно использовать механизм рекомендаций.

Если кто-то хочет поделиться Визиткой с другим пользователем, то он может отправить ему Визитку:

а) Если у получателя тоже есть аккаунт AdveCard, то он получит рекомендуемую Визитку и сможет добавить к себе в Личную Визитницу, при этом она попадает в сообщения, откуда пользователь может сохранить ее в свою Персональную Визитницу.

Если пользователь рекомендует Приватную Визитку, то перед тем как Визитка попадет адресату, у владельца Визитки запрашивается разрешение, если владелец даст свое разрешение, то адресат получит Визитку, если запретит, то рекомендация отменяется.

б) Если аккаунта нет, то пользователь может выбрать приложение отправки, например WhatsApp, Telegram, любой другой мессенджер или посредством СМС. При этом получателю придет ссылка, по которой можно открыть Визитку через браузер.

Отметка «Я рекомендую»

Для оценки специалистов применяется система рекомендаций. Пользователи могут на визитке специалиста оценить его работу с помощью кнопки «Я рекомендую» (подобно Лайкам на фейсбук) .

Количество рекомендаций составляют рейтинг Визитки.

Тот, кто поставил отметку «Рекомендовать», может также отменить эту отметку.

1.8 Встречи

Встреча – это запись в системе, которая идентифицирует физическую встречу и позволяет соединить несколько девайсов для обмена визитками. Пользователи могут создавать Встречи в приложении и распространять их.

Встречу можно создать на бесконечное время или на определенный промежуток, по истечении которого Встреча автоматически закроется.

При создании Встречи формируется Код встречи и ее графическое представление в виде QR-code.

QR-код включает в себя ссылку на страницу, открывающуюся в браузере, в которой отображается передаваемая Визитка.

К каждой встрече пользователи должны прикрепить одну из своих Визиток, которую нужно передать на данной встрече.

Если Встреча была создана как приватная, то инициатор Встречи должен подтвердить участника. После чего любой из них может послать друг другу свои Визитки.

Другие пользователи, чтобы прикрепиться к встрече могут или отсканировать через приложение QR-code или ввести код Встречи вручную.

Возможны несколько вариантов присоединения к Встрече

а) Двое или несколько пользователей проводят физическую встречу и могут общаться непосредственно. В данном случае один из встречающихся должен создать Встречу. Остальные участники подключаются к ней. После этого аккаунты всех присутствующих будут соединены, после чего можно сделать обмен визитками.

б) Пользователи находятся в одном месте, могут видеть друг друга, но личного контакта нет, например, во время конференции выступающий не может поговорить лично с каждым слушателем. При невозможности личного контакта, инициатор создает Встречу заранее и распечатывает QR-code крупным шрифтом и размещает на видном месте перед входом и/или в зале, что позволит каждому слушателю отсканировать код встречи на расстоянии. Заведения могут выставлять визитки на видных местах или размещать их в своих рекламных средствах.

в) Пользователи не встречаются физически, но имеют номера телефонов друг друга у себя на смартфонах в контактах. В этом случае приложение позволяет скинуть визитку, указав номер из своих контактов;

г) Пользователи не встречаются физически, и не имеют номера телефонов в контактах друг друга.

Если пользователи звонят друг другу, то можно отправить визитку на номер входящего/исходящего звонка, если они переписываются через мессенджеры, соц. сети или email, то они могут послать друг другу ссылки на визитки.

В каждый момент времени аккаунт может быть подключен только к одной Встрече. Если пользователь присоединяется к другой встрече, то контакт по старой Встрече отключается.

1.9 Передача визитки

Чтобы отправить Визитку другому пользователю нужно открыть свою визитку и нажать кнопку «Отправить», после этого откроется окно выбора вида передачи: по подключенной Встрече, по номеру телефона из Контактов. После выбора типа передачи Визитка будет отправлена в аккаунт адресата, или, если

контакт не использует AdveCard, то ссылка на Визитку будет отправлена адресату на его предпочитаемый мессенджер или СМС-кой.

При отправке Визитки другому пользователю AdveCard в сообщениях будут отображаться рекомендуемые Визитки. Откуда Визитку можно переместить в свою Визитницу или просто удалить как не нужную.

Если у адресата приложение AdveCard не установлено, то ему можно отправить ссылку на визитку через любой мессенджер или посредством СМС.

Лента сообщений.

Все сообщения пользователям в системе AdveCard отображаются в специальном разделе (во вкладке) внешне схожим с обычной лентой сообщений мессенджеров.

В данной ленте должны отображаться: запросы на контактные данные, новости приложения, системный сообщения и переписка с администраторами AdveCard.

Обратная связь.

Каждый пользователь, столкнувшись с проблемами в системе или правообладания аккаунтом или визитки, может написать в техническую поддержку системы сообщение через специальную форму в веб-приложении AdveCard.

2 Описание используемых средств разработки

2.1 PostgreSQL – объектно-реляционная СУБД (система управления базами данных)

PostgreSQL – это бесплатно распространяемая объектно-реляционная СУБД (ORDBMS), одна из самых функциональных открытых систем управления БД в мире и являющаяся одной из самых сильных альтернатив платным базам данных.

PostgreSQL разработана на базе некоммерческой СУБД Postgres, созданной как open-source проект в Беркли (университет в Калифорнии). Разработка Postgres началась в 1986 году, к её созданию имел прямое отношение Майкл Стоунбрейкер, создатель более раннего проекта Ingres, которую на тот момент уже приобрела компания Computer Associates. Название расшифровывается как «Post Ingres», и при разработке Postgres программисты применили большинство уже ранее сделанных наработок.

Стоунбрейкер со своими студентами начали разработку Postgres в 1986 году, но первая версия вышла только в 1994 году. За это время в Postgres были введены правила, пользовательские типы, процедуры и другие компоненты. Разработка БД разделилась в 1995 году: Стоунбрейкер использовал свой прошлый опыт в разработке коммерческой СУБД Illustra, развиваемой его личной софтверной компанией (которую впоследствии купила компания Informix), а его ученики создали Postgres95 – новую версию Postgres, заменив в ней язык запросов POSTQUEL (использовался в Ingres) – на SQL.

Создание Postgres95 было выведено за пределы Калифорнийского университета и передана команде молодых разработчиков-энтузиастов. Новая база данных получила название, под которым она сейчас известна и активно развивается — PostgreSQL.

Функции

Функции – это блоки кода, которые исполняются не на клиентском устройстве, а на сервере. Функции можно писать на чистом SQL, но иногда необходим такой функционал, который невозможно реализовать только лишь на SQL, например, использование циклов, для этого приходится использовать дополнительные языковые расширения. Функции можно писать с использованием следующих языковых расширений:

В PostgreSQL разрешены функции, которые могут возвращать набор записей, который используется также, как и результат выполнения обыкновенного запроса.

Таблицы могут наследовать характеристики и наборы полей от других таблиц (родительских). При этом данные, добавленные в дочернюю таблицу,

автоматически будут участвовать (если это не указано отдельно) в запросах к родительской таблице.

В PostgreSQL 10 был добавлен механизм партицирования таблиц. Партицирование необходимо для деления одной таблицы на несколько, которые называемые партиции. Партицирование похоже на наследование, но имеет более простой синтаксис и более строгие ограничения, что позволяет производить дополнительную оптимизацию при планировании запросов.

2.2 Apache Tomcat + Java сервлеты – сервлеты для обработки запросов системы на сервере

Сервлет является интерфейсом Java, реализация которого расширяет функциональные возможности сервера. Сервлет взаимодействует с клиентами посредством принципа запрос-ответ. Хотя сервлеты могут обслуживать любые запросы, они обычно используются для расширения веб-серверов. Для таких приложений технология Java Servlet определяет HTTP-специфичные сервлет классы. Пакеты javax.servlet и javax.servlet.http обеспечивают интерфейсы и классы для создания сервлетов.

Контейнер сервлетов – программа, представляющая собой сервер, который занимается системной поддержкой сервлетов и обеспечивает их жизненный цикл в соответствии с правилами, определёнными в спецификациях.

Известные реализации: Apache Tomcat, Jetty, JBoss, GlassFish, IBM WebSphere, Oracle Weblogic.

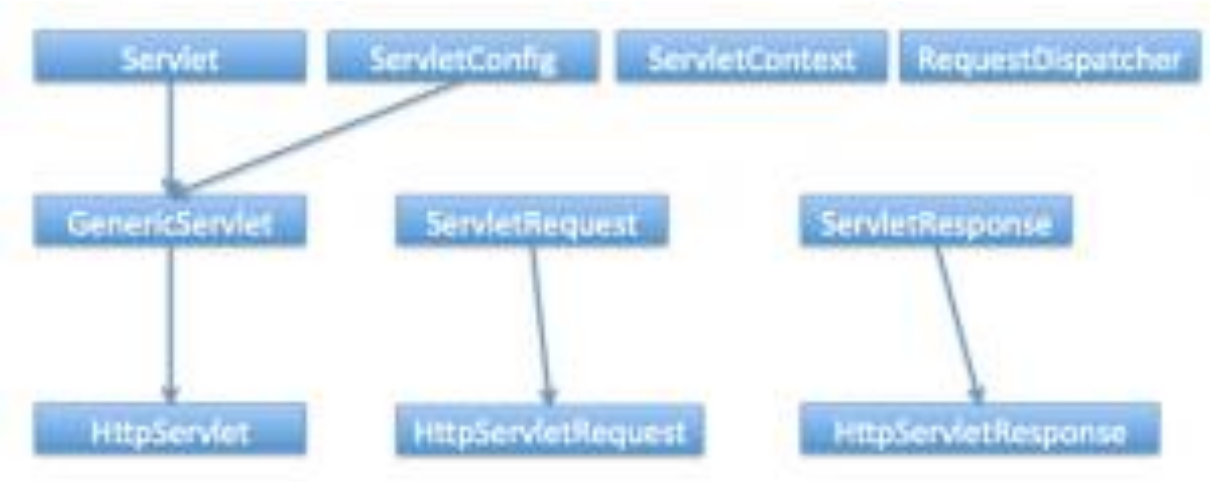


Рисунок 2.1 – Структура сервлета

Контейнер сервлетов может работать как полноценный самостоятельный веб-сервер, быть поставщиком страниц для другого веб-сервера, например

Apache, или интегрироваться в Java EE сервер приложений. Обеспечивает обмен данными между сервлетоми клиентами, берёт на себя выполнение таких функций, как создание программной среды для функционирующего сервлета, идентификацию и авторизацию клиентов, организацию сессии для каждого из них.

Сервлетный фильтр, в соответствии со спецификацией, это Java-код, пригодный для повторного использования и позволяющий преобразовать содержание HTTP-запросов, HTTP-ответов и информацию, содержащуюся в заголовках HTTP. Сервлетный фильтр занимается предварительной обработкой запроса, прежде чем тот попадает в сервлет, и/или последующей обработкой ответа, исходящего из сервлета.Сервлетные фильтры могут:

- перехватывать инициализацию сервлета прежде, чем сервлет будет инициирован;
- определить содержание запроса прежде, чем сервлет будет инициирован;
- модифицировать заголовки и данные запроса, в которые упаковывается поступающий запрос;
- модифицировать заголовки и данные ответа, в которые упаковывается получаемый ответ;
- перехватывать инициализацию сервлета после обращения к сервлету.

Сервлетный фильтр может быть сконфигурирован так, что он будет работать с одним сервлетом или группой сервлетов. Основой для формирования фильтров служит интерфейс javax.servlet.Filter, который реализует три метода:

void init (FilterConfig config) throws ServletException;

void destroy();

void doFilter (ServletRequest request, ServletResponse response, FilterChain chain) throws IOException, ServletException;

Метод init() вызывается прежде, чем фильтр начинает работать,и настраивает конфигурационный объект фильтра. Метод doFilter выполняет непосредственно работу фильтра. Таким образом, сервер вызывает init() один раз, чтобы запустить фильтр в работу, а затем вызывает doFilter() столько раз, сколько запросов будет сделано непосредственно к данному фильтру. После того, как фильтр заканчивает свою работу, вызывается метод destroy().

Т.к. браузер понимает только HTTP, то когда приложение выбросит исключение контейнер сервлетов обработает исключение и создаст HTTP response. Это аналогично тому что происходит при кодах ошибок вроде 404, 403 и т.д. Servlet API предоставляет поддержку собственных сервлетов для обработки исключений и ошибок, которые мы можем задать в дескрипторе развертывания. Главная задача таких сервлетов — обработать ошибку или исключение и отправить понятный HTTP ответ пользователю. Например, можно предоставить ссылку на главную страницу, а также описание некоторых деталей об ошибке.

Контейнер сервлетов обычно загружает сервлет при первом запросе клиента, но иногда необходимо загрузить сервлет прямо на старте приложения (например если сервлет объемный и будет долго грузиться). Для этого необходимо использовать элемент load-on-startup в дескрипторе (или аннотацию loadOnStartup), который укажет необходимость загрузки сервлета при запуске.

Значение должно быть int. Если значение отрицательное, то сервлет будет загружен при запросе клиента, а если 0 и далее, то загрузится на старте приложения. Чем меньше число, тем раньше в очереди на загрузку будет сервлет.

Контейнер сервлетов управляет четырьмя фазами жизненного цикла сервлета:

- Загрузка класса сервлета — когда контейнер получает запрос для сервлета, то происходит загрузка класса сервлета в память и вызов конструктора без параметров.
- Инициализация класса сервлета — после того как класс загружен контейнер инициализирует объект ServletConfig для этого сервлета и внедряет его через init() метод. Это и есть место где сервлет класс преобразуется из обычного класса в сервлет.
- Обработка запросов — после инициализации сервлет готов к обработке запросов. Для каждого запроса клиента сервлет контейнер порождает новую нить (поток) и вызывает метод service() путем передачи ссылки на объект ответа и запроса.
- Удаление из Service — когда контейнер останавливается или останавливается приложение, то контейнер сервлетов уничтожает классы сервлетов путем вызова destroy() метода.

Можно описать как последовательность вызова методов: init(), service(), destroy().

- public void init(ServletConfig config) – используется контейнером для инициализации сервлета. Вызывается один раз за время жизни сервлета.
- public void service(ServletRequest request, ServletResponse response) – вызывается для каждого запроса. Метод не может быть вызван раньше выполнения init() метода.
- public void destroy() – вызывается для уничтожения сервлета (один раз за время жизни сервлета).

В данном дипломном проекте в качестве контейнера сервлетов был выбран Apache Tomcat. Apache Tomcat — opensource-проект, который реализует спецификацию контейнера сервлетов и спецификацию JavaServer Pages (JSP). Используется в качестве самостоятельного сервера веб-приложений, в качестве сервера контента в связке с веб-сервером Apache, а также в качестве контейнера сервлетов в серверах приложений JBoss и GlassFish.

Apache Tomcat сервер состоит из трех основных компонентов: Jasper (механизм JSP), Catalina (контейнер сервлетов) и Coyote (стек HTTP):

- Jasper — механизм JSP Tomcat'a, который является реализацией спецификации JavaServer Pages 2.0 Sun Microsystems. Jasper анализирует JSP-файлы, чтобы компилировать их в Java код, как сервлеты (которые могут быть обработаны с помощью Catalina). Во время выполнения, Jasper может автоматически обнаруживать изменения JSP-файла и перекомпилировать его.

- Catalina — контейнер сервлетов Tomcat'a, который реализует спецификацию Servlet API. Servlet API является основой для всех остальных технологий Java, касающихся Web и дает возможность динамически генерировать любой web-контент, используя любые библиотеки, доступные для java.

- Coyote — компонент стека HTTP Tomcat'a, который поддерживает протокол HTTP для веб-серверов или контейнера приложений. Coyote прослушивает входящие соединения на определённом TCP порту сервера, пересылает запросы в механизм Tomcat для обработки запросов и отправляет ответ назад запрашивающему клиенту.

Установка Apache Tomcat происходит следующим образом:

Сначала нужно обновить систему.

$ sudo apt-get update

Затем установить Java Development Kit. Иначе код Java не будет выполняться.

$ sudo apt-get install default-jdk

Установка Tomcat версии 8 и других дополнений Tomcat.

$ sudo apt-get install tomcat8

$ sudo apt-get nstall tomcat8-docs tomcat8-examples tomcat8-admin

Запустить Tomcat8.

$ sudo systemctl start tomcat8

Файлы настроек XML

Два самых важных файла настроек, чтобы запустить Tomcat, называются .xml и web.xml. По умолчанию они размещены в TOMCAT-HOME/conf/server.xml и TOMCAT-HOME/conf/web.xml, соответственно.

SERVER.XML

Файл server.xml — главный файл настроек Tomcat. Элементы server.xml относятся к пяти базовым категориям:

- Элементы верхнего уровня (Top Level Elements);
- Соединители или коннекторы (Connectors);
- Контейнеры (Containers);
- Встраиваемые компоненты (Nested Components);
- Глобальные настройки (Global Settings).

У всех элементов из этих категорий имеется множество атрибутов, которые позволяют точно определить функциональные возможности. Чаще всего если необходимо внести какие-то существенные изменения в установку Tomcat, как, например, изменить число портов, приходится редактировать файл server.xml.

На сайте Apache's Tomcat Documentation содержится достаточно информации, но нет некоторых сведений о настройках элементов.

Элементы верхнего уровня:

- Server (сервер). Этот элемент определяет отдельный сервер Tomcat и содержит элементы конфигурации Logger и ContextManager. К тому же, элемент Server поддерживает атрибуты "port", "shutdown" и "className". Атрибут порт используется для того, чтобы уточнить, через какой порт должны выполняться команды shutdown (отключения). Атрибут shutdown задает командную строку для отдельного порта, чтобы спровоцировать отключение. Атрибут className — реализацию класса Java, которая должна использоваться.

- Service (сервис). Это элемент, который можно поместить в элемент Server; он содержит один или несколько компонентов Connector, у которых один общий компонент Engine. Главная функция этого компонента — задать эти компоненты как один сервис. Название сервиса, который будет появляться в логах, определяется с помощью атрибута "name" (элемент Service).

- Connectors (соединители). Размещая один или несколько соединителей (connector) в теге Service, вы тем самым позволяете системе перенаправить запросы из этих портов в один компонент Engine для обработки. Tomcat позволяет определить соединители HTTP и AJP.

- HTTP- соединитель. Этот элемент представляет HTTP/1.1 Connector и обеспечивает Catalina автономным функционалом веб-сервера. Это означает, что в дополнение к выполнению сервелатов и JSP -страниц, Catalina способен прослушивать специфические TCP-порты для запросов.

- AJP-соединитель. Данный элемент является соединителем, который обеспечивает связь с протоколом AJP. Главная роль элемента в том, чтобы помочь Tomcat работать в связке с Apache.

- Контейнеры. С помощью этих элементов Catalina направляет запросы в корректный обрабатывающий аппарат.

- Context. Этот элемент представляет определенное веб-приложение и содержит данные о пути, по которому определяются запросы для соответствующих ресурсов приложения. Catalina получает запрос и пытается сопоставить самый длинный URI с контекстным путем определенного элемента Context до тех пор, пока не найдется корректный элемент, который бы обслуживал запрос.

У элемента Context может быть максимум один встроенный экземпляр на один элемент из вспомогательных элементов Loader, Manager, Realm, Resources и WatchedResource.

Хотя Tomcat позволяет определять элементы Context в “TOMCAT-HOME/conf/server.xml”, этого лучше избегать, поскольку эти главные настройки нельзя перезагрузить без перезагрузки Tomcat.

- Engine. Этот элемент используется в связке с одним или несколькими соединителями, которые размещены в элементе Service. Элемент Engine может использоваться только в случае если он размещен в элементе Service, и только один элемент Engine разрешен в элементе Service. Обращайте внимание на атрибут “defaultHost”, который задает элемент Host. Последний отвечает за обслуживание запросов для названий хостов на сервере, которые не настраиваются в server.xml. Название этого атрибута должно совпадать с названием одного из элементов Host, которые размещены в элементе Engine. Также важно выбрать уникальное, логичное название для каждого из элементов Engine, используя атрибут “name”. Если один элемент Server в вашем файле server.xml включает несколько элементов Service, потребуется выбрать уникальное название для каждого элемента Engine.

- Host. Элемент, который размещен в элементе Engine, и используется, чтобы связать названия серверной сети с серверами Catalina . Этот элемент будет функционировать должным образом только если виртуальный хост был зарегистрирован в системе DNS соответствующего домена. Одна из самых полезных особенностей элемента Host — возможность содержать элементы Alias, использующиеся для того, чтобы определить названия нескольких сетей.

- Cluster. Элемент Cluster используется в Tomcat для того, чтобы обеспечить репликацию контекстных атрибутов, использование WAR, репликацию сессий, и может размещаться в элементе Engine или Host. В них размещаются элементы Manager, Channel, Valve, Deployer и ClusterListener. Больше информации об этих элементах и о том, как они используются, можно найти на странице Apache Tomcat Configuration. И хотя у этого элемента множество различных конфигураций, стандартной конфигурации будет вполне достаточно, чтобы учесть интересы большинства пользователей при разработке приложения.

- Nested Components. Эти элементы размещаются внутри элементов container, чтобы задать дополнительные функциональные возможности.

- Listeners. Эти элементы можно поместить внутрь элементов Server, Engine, Host или Context. Они указывают на компонент, который производит определенное действие при специфическом событии.

У большинства компонентов есть атрибуты className, чтобы выбрать разные реализации элемента. Существует ряд дополнительных реализаций

Listener, не только дефолтных. Все эти реализации требуют, чтобы элемент Listener размещался в определенном элементе Server.

И крайне важно настроить этот атрибут корректно. Доступные на текущий момент реализации содержатся в APR Lifecycle Listener, Jasper Listener, Server Lifecyle Listener, Global Resources Lifecyle Listener, JMX Remote Lifecycle Listener и в JRE Memory Leak Prevention Listener.

- Global Naming Resources. Этот элемент используется, чтобы определить ресурсы Java Naming and Directory Interface для специфического Server, отличного от любых контекстов веб-приложения JNDI. Если нужно, вы можете задать характеристики JNDI resource lookup для <resource-ref> и <resource-env-ref> в данном элементе, определив их и связав с помощью <ResourceLink>.

Результаты этого метода эквивалентны добавлению элементов <resource-ref> в файл приложения “/WEB-INF/web.xml”. Если используете эту технику, проверьте, что вы определили дополнительные параметры, которые необходимы, чтобы задать и настроить объект-фабрику и свойства.

- Realm. Этот элемент размещается в любом элементе Container и задает базу данных, содержащую имена пользователей, пароли и роли для Container. При размещении внутри элемента Host или Engine, характеристики, заданные в элементе Realm, передаются всем контейнерам нижнего уровня по умолчанию.

Важно корректно установить атрибут “className” этого элемента, поскольку существует множество реализаций. Эти реализации используются, чтобы сделать доступным Catalina другим системам управления безопасностью пользователей (например, JDBC, JNDI или DataSource).

- Resources. У этого элемента только одно предназначение – направить Catalina в статические ресурсы, которые используются вашими веб-приложениями. Эти ресурсы включают классы, HTML и JSP файлы. Использование этого элемента предоставляет Catalina доступ к файлам, содержащимся в других местах, помимо файловой системы (filesystem), таким как ресурсы, которые содержатся в архивах WAR или базах данных JDBC.

Важно помнить, что это техника предоставления веб-приложениям доступа к ресурсам, которые содержатся вне файловой системы, может использоваться только, если приложение не требует непосредственного доступа к ресурсам, которые хранятся в файловой системе.

- Valve. Компоненты Valve размещаются внутри элементов Engine, Host и Context, с их помощью добавляются специальные функциональные возможности в конвейер, обрабатывающий запросы. Это очень разносторонний элемент. Существует множество различных типов элементов Valve – от аутентификаторов до фильтров и исправлений ошибок WebDAV. Многие из этих типов Valve размещаются только внутри специальных элементов.

- Web.XML. Файл web.xml содержит информацию, которая используется для конфигурации компонентов ваших веб-приложений. Задавая конфигурацию Tomcat в первый раз, вы можете задать servlet-mapping для центральных компонентов, таких как JSP. В Tomcat этот файл функционирует так же, как описано в спецификации Servlet.

Единственное отличие в том, как Tomcat обрабатывает этот файл: есть опция задать с помощью TOMCAT-HOME/conf/web.xml значения по умолчанию для всех контекстов. Если используется такой метод, базовой конфигурацией будет служить TOMCAT-HOME/conf/web.xml, который может переписать специфические для приложения файлы WEB-INF/web.xml.

2.3 Joomla – система управления веб-проектом

Joomla! – система управления веб-порталом (CMS), написанная на языках PHP и JavaScript, которая использует в качестве хранилища базы данных СУБД MySQL или многие другие реляционные СУБД. Является свободным программным обеспечением, распространяемым под лицензией GNU GPL.

Joomla! написана на PHP, она использует методы объектно-ориентированного программирования (начиная с версии 1.5) и паттерны проектирования программного обеспечения, хранит данные в базе данных MySQL, MS SQL (начиная с версии 2.5) или PostgreSQL (начиная с версии 3.0) и включает в себя такие функции, как кеширование страниц, RSS-каналы, печатные версии страниц, новости, блоги, поиск и поддержка интернационализации языка.

По состоянию на май 2018 года, Joomla! была загружен более 93 миллионов раз. Более 8 000 бесплатных и коммерческих расширений доступны на официальном сайте Joomla! Extensions Directory и других свободно доступных источниках. По оценкам статистиков, это вторая наиболее используемая система управления контентом в Интернете после WordPress.

Joomla! Стала ответвлением другой CMS Mambo 17 августа 2005 г. Разработчики Joomla! создали сайт под названием OpenSourceMatters.org (OSM) для распространения информации среди сообщества разработчиков программного обеспечения. Руководитель проекта Эндрю Эдди написал письмо, которое появилось в разделе объявлений общественного форума на сайте mamboserver.com. Более тысячи человек присоединились к OpenSourceMatters.org в течение дня, большинство из которых высказывали слова поддержки. В результате сайт получил эффект сарафанного радио.

18 августа Эндрю Эдди обратился к сообществу с просьбой предложить название проекта. Основная команда зарезервировала право на окончательное решение об именах и выбрала имя, не рекомендованное сообществом. 22 сентября было объявлено новое имя, Joomla. Это англизированное написание слова на

суахили jumla, означающее все вместе или в целом, которое также имеет аналогичное значение, по крайней мере, в амхарском, арабском и

Как и многие другие веб-приложения, Joomla! может выполняться в стеке LAMP.

Многие веб-хосты имеют панели управления для автоматической установки Joomla. В Windows Joomla можно установить с помощью установщика Microsoft Web Platform, который автоматически обнаруживает и устанавливает зависимости, такие как PHP или MySQL.

Миграция / configuration.php

Joomla! использует конфигурационный файл (configuration.php, обычно расположенный в корневом каталоге установки Joomla!) для управления различными настройками, включая (но не ограничиваясь) настройками подключения к базе данных. Из-за использования файла конфигурации миграция с одного сервера на другой относительно проста.

Для добавления нового функционала Joomla! Использует расширения. Можно выделить восемь типов расширений: компоненты, модули, плагины, шаблоны, языки, библиотеки, файлы и пакеты. Каждое из этих расширений обрабатывает определенную функцию.

Компоненты являются самыми большими и сложными расширениями. Большинство компонентов имеют две части: пользовательскую часть и часть администратора. Каждый раз, когда Joomla! загружает страницы, один компонент вызывается для отображения основного тела страницы. Компоненты производят основную часть страницы, потому что компонент управляется элементом меню.

Плагины - это расширенные расширения и, по сути, обработчики событий. Когда событие запускается, плагины, зарегистрированные для обработки этого события, выполняются. Например, плагин может использоваться для блокирования статей, отправленных пользователем, и фильтрования текста. Линия между плагинами и компонентами иногда может быть немного нечеткой. Иногда большие или продвинутые плагины называются компонентами, даже если они фактически не отображают большую часть страницы.

Шаблоны описывают основной дизайн Joomla!. Пока CMS управляет содержимым веб-сайта, шаблоны определяют стиль, внешний вид и расположение всех блоков сайта.

Модули являются динамическим или статическим выводом информации в определенные позиции шаблона. Шаблоны определяют динамические позиции, которым могут быть присвоены модули. Примером может быть форма входа на боковой панели. Это можно сравнить с другими виджетами CMS на боковой панели. В каждую позицию можно назначить несколько модулей, и каждое назначение модуля может контролироваться в каждом элементе меню.

Языки – это очень простые расширения, которые могут использоваться как основная часть или как расширение.

Библиотеки обычно представляют собой дополнительные php-библиотеки, которые обеспечивают функциональность для правильного функционирования компонента, модуля или плагина (например, API Google).

Файлы - это одиночные файлы, которые можно установить в любом месте файловой системы Joomla!. Примеры этого включают предоставление разработчикам расширений для предоставления дополнительных представлений шаблонов.

Пакеты позволяют пользователю устанавливать комбинации любого другого типа расширения, указанного выше. Это позволяет устанавливать и удалять связанные пакеты в одном действии, а не как отдельные объекты.

3 Практическая часть

3.1 Установка и настройка сервера

Для стабильной и бесперебойной работы всей системы было принято решение не использовать стандартный виртуальный хостинг, а взять в аренду сервер и установить его в дата-центре компании PS.kz. PS.kz – лидер на рынке предоставления услуг хостинга и покупки доменов в Республике Казахстан. Компания «Web Line» успешно сотрудничает с PS.kz на протяжении пяти лет, за это время не возникало никаких проблем, а все рабочие вопросы решались в короткие сроки сотрудниками PS.kz. Ниже, на рисунке 3.1 представлена панель управления выделенным сервером:

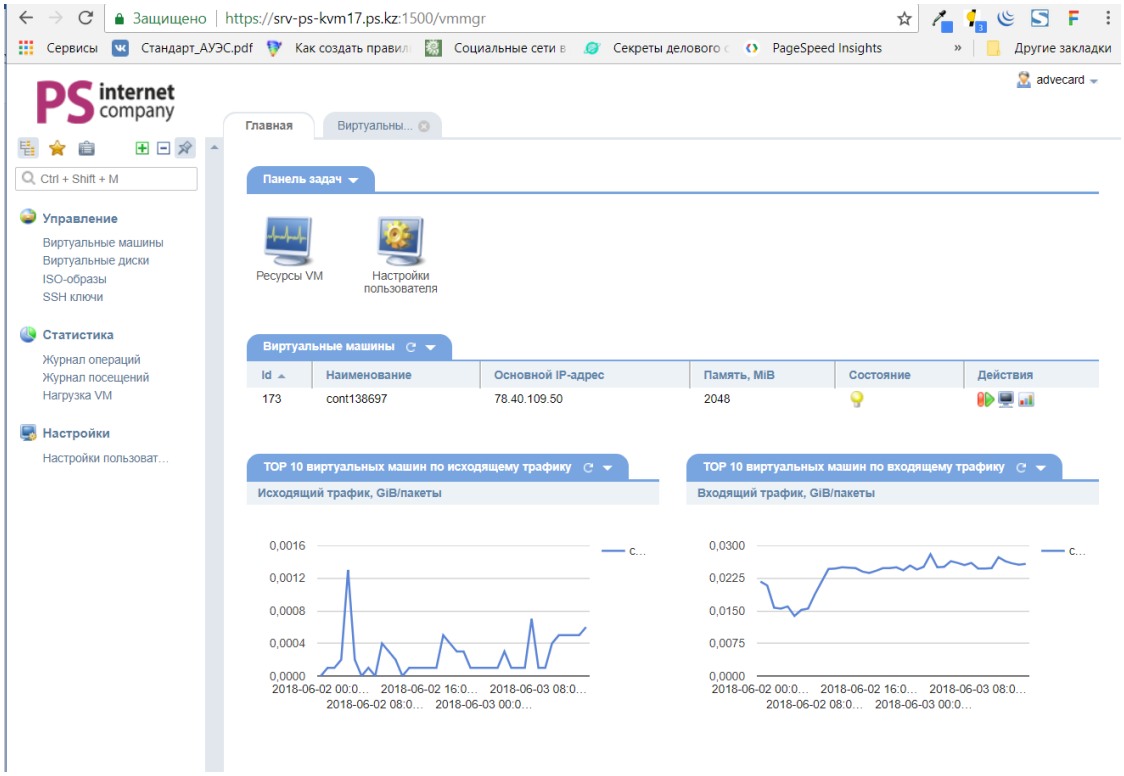


Рисунок 3.1 – Панель управления выделенным сервером PS.kz

В качестве операционной системы на сервере было принято решение установить CentOS 7.0. CentOS – дистрибутив Linux, основанный на коммерческом Red Hat Enterprise Linux компании Red Hat и совместимый с ним. Срок поддержки каждой версии CentOS составляет 10 лет. Каждая версия обновляется каждые 6 месяцев для поддержки новых аппаратных средств.

CentOS - это полноценная операционная система, которая не является упрощённой низкопробной копией. Это самостоятельный проект, который, тем не менее, имеет общий базовый программный код с творением Red Hat Enterprise. Сразу нужно дать одно важное пояснение. Это не пиратская версия, а вполне легальная система. Если говорить более предметно, вся соль заключается в том, что Red Hat по собственному желанию выкладывают в открытый доступ исходные коды. Red Hat Enterprise Linux состоит из свободного ПО с открытым кодом, но доступен в виде дисков с бинарными пакетами только для платных подписчиков. Как требуется в лицензии GPL и других, Red Hat предоставляет все исходные коды. Разработчики CentOS используют данный исходный код для создания окончательного продукта, очень близкого к Red Hat Enterprise Linux и доступного для загрузки. Существуют и другие клоны Red Hat Enterprise Linux, созданные на основе этого кода. Операционная система разработана энтузиастами, тем не менее, она имеет постоянные обновления. На данный момент последняя седьмая версия включает полный пакет всех нужных нововведений в сфере защиты. Новые версии выпускаются раз в два года, пакет обновлений каждые полгода.

Начиная с Red Hat Enterprise Linux 7, инфраструктура проекта предоставляется Redhat, а исходный код Red Hat Enterprise Linux 7 перемещен с основного сервера на новые серверы CentOS. Теперь на ftp.redhat.com содержится вместо исходного кода ссылка на исходный код, переданный непосредственно CentOS.

CentOS была выбрана по ряду причин, перечисленных ниже. Centos – беслпатный дистрибутив в отличие от той же RHEL, которая предоставляется на коммерческой основе. CentOS имеет более сложный инсталлятор, чем большинство Linux-дистрибутивов, но в нём доступно гораздо большее количество параметров. Со стороны технических аспектов выделяют оперативность репозиториев RHEL на высоком уровне, чем обеспечивается безопасность системы. Используются технологии GCC как SSP (защита стека), PIE. Набор ПО актуальный и типичный для современных ОС: предоставляются версии офисных, серверных и девелоперских пакетов, программ и утилит (KDE и Gnome с compiz и AIGLX, Firefox и Evolution, MySQL и PostgreSQL, Apache и PHP, и т.д.). CentOS проходит тщательную проверку и тестирование перед выходом в свет. Да, в этом дистрибутиве установлена более старая версия программного обеспечения, но всё тщательно подогнано, настроено и проверено. Также предоставляется подробная техническая документация и имеется большой штат поддержки ОС, к которому можно обратится и получить ответы на все интересующие вопросы по данной системе. Для установки была выбрана версия без графического интерфейса, к которой можно подключаться через VNS клиент (Virtual Network Computing), который представлен на Рисунке 3.2:

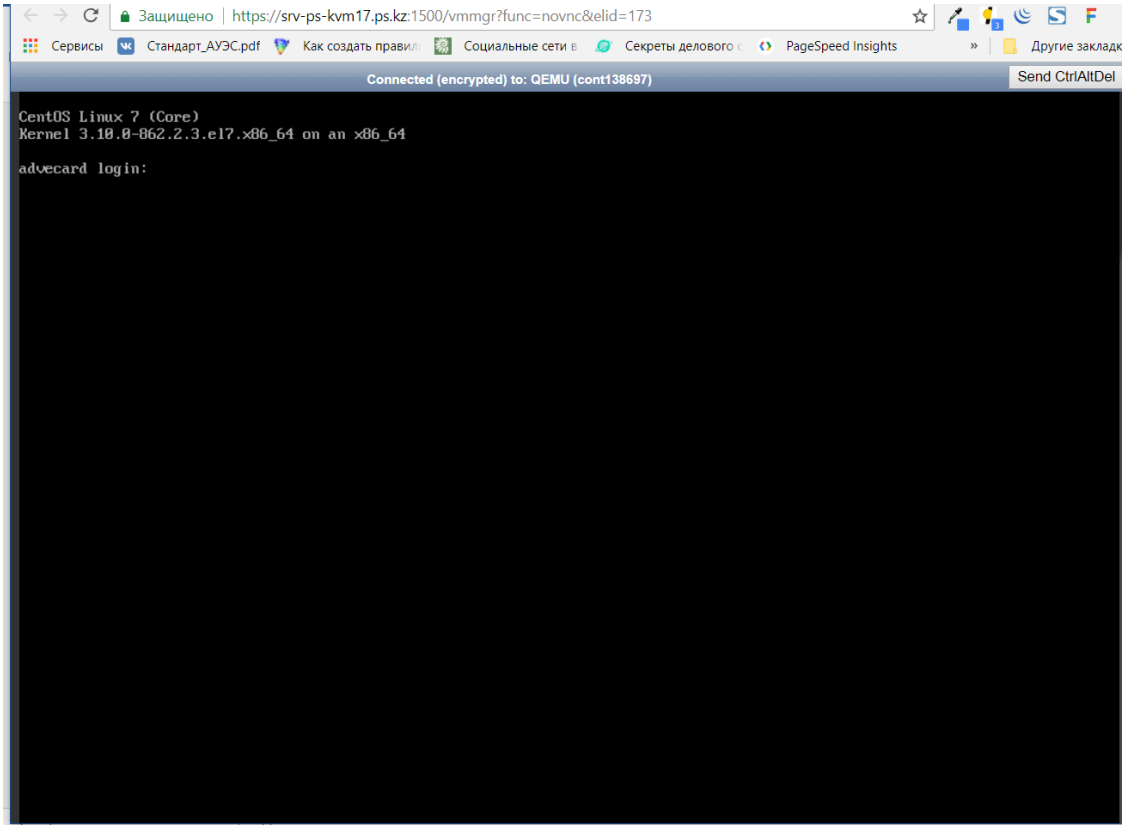


Рисунок 3.2 – Доступ к выделенному серверу через VNC клиент

3.2 Установка и настройка контейнера сервлетов Apache Tomcat

Apache Tomcat (ранее — Catalina) — это контейнер сервлетов с открытым исходным кодом, разрабатываемый Apache Software Foundation. Реализует спецификацию сервлетов и спецификацию JavaServer Pages (JSP) и JavaServer Faces (JSF). Написан на языке Java. Томкат дает возможность запускать приложения для веба, которые содержат ряд утилит для самоконфигурирования.

Tomcat используется в качестве самостоятельного веб-сервера, в качестве сервера контента в сочетании с веб-сервером Apache HTTP Server, а также в качестве контейнера сервлетов в серверах приложений JBoss и GlassFish. Apache Tomcat используется для развертывания ваших сервлетов Java и JSP. Таким образом, в вашем проекте Java вы можете создать свой файл WAR (short for Web ARchive) и просто поместить его в каталог развертывания в Tomcat. Таким образом, в основном Apache является HTTP-сервером, обслуживающим HTTP. Tomcat - сервер Servlet и JSP, обслуживающий Java-технологии.

Tomcat - контейнер сервлетов. Сервлет, в конце, является классом Java. Файлы JSP (похожие на PHP и более старые файлы ASP) генерируются в Java-код (HttpServlet), который затем скомпилируется в файлы .class сервером и

выполняется виртуальной машиной Java. Для установки Tomcat необходимо зайти через VNC клиент на сервер и выполнить следующие команды:

wget http://www.us.apache.org/dist/tomcat/tomcat-8/v8.0.14/bin/apache-tomcat-8.0.14.tar.gz -O tomcat8.tar.gz

```
tar -xvf tomcat8.tar.gz
mv apache-tomcat-* /opt/tomcat8

nano /home/serveradmin/.bashrc
export CATALINA_HOME=/opt/tomcat8

nano /etc/init.d/tomcat8
#!/bin/bash

export CATALINA_HOME=/opt/tomcat8
export JAVA_OPTS="-Xms1024M -Xmx1024M -Dfile.encoding=UTF-8";

PATH=/sbin:/bin:/usr/sbin:/usr/bin
start() {
 sh $CATALINA_HOME/bin/startup.sh
}
stop() {
 sh $CATALINA_HOME/bin/shutdown.sh
}
case $1 in
  start|stop) $1;;

  restart) stop; start;;
  *) echo "Run as $0 &lt;start|stop|restart&gt;"; exit 1;;
esac

chmod 755 /etc/init.d/tomcat8
/etc/init.d/tomcat8 start

/etc/init.d/tomcat8 restart
/etc/init.d/tomcat8 stop

nano /etc/rc.local
/etc/init.d/tomcat8 start
```

Интерфейс Apache Tomcat показан на Рисунке 3.3:

Tomcat Web Application Manager

Message:	OK

Manager			
List Applications	HTML Manager Help	Manager Help	Server Status

Applications

Path	Version	Display Name	Running	Sessions	Commands
/	*None specified*	Welcome to Tomcat	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/AdveCardAdmin	*None specified*		true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/docs	*None specified*	Tomcat Documentation	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/examples	*None specified*	Servlet and JSP Examples	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/host-manager	*None specified*	Tomcat Host Manager Application	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/manager	*None specified*	Tomcat Manager Application	true	1	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes

Deploy

Deploy directory or WAR file located on server

Context Path (required):

XML Configuration file URL:

WAR or Directory URL:

Deploy

WAR file to deploy

Select WAR file to upload Выберите файл Файл не выбран

Deploy

Diagnostics

Check to see if a web application has caused a memory leak on stop, reload or undeploy

Find leaks	This diagnostic check will trigger a full garbage collection. Use it with extreme caution on production systems.

TLS connector configuration diagnostics

Ciphers	List the configured TLS virtual hosts and the ciphers for each.
Certificates	List the configured TLS virtual hosts and the certificate chain for each.
Trusted Certificates	List the configured TLS virtual hosts and the trusted certificates for each.

Server Information

Tomcat Version	JVM Version	JVM Vendor	OS Name	OS Version	OS Architecture	Hostname	IP Address
Apache Tomcat/8.5.31	1.8.0_171-b11	Oracle Corporation	Linux	3.10.0-862.2.3.el7.x86_64	amd64	advecard.kz	78.40.109.50

Рисунок 3.3 – Интерфейс Apache Tomcat 8

3.3 Установка и настройка CMS Joomla

Для установки и корректного использования CMS Joomla сервер должен удовлетворять требованиям, указанным в Таблице 3.1.

Таблица 3.1 – Минимальные и рекомендуемые требования для CMS Joomla

Программное обеспечение	Рекомендовано	Минимум	Дополнительная информация
PHP (Magic Quotes GPC, MB String Overload = off / Zlib Compression Support, XML Support, INI Parser Support, JSON Support, Mcrypt Support, MB Language = Default)	5.6 или 7 и следующие за ней версии	5.3.10 +	www.php.net
Поддерживаемые базы данных:			
MySQL (Требует поддержки InnoDB)	5.5.3 +	5.1 +	www.mysql.com
SQL Server	10.50.1600.1 +	10.50.1600.1 +	www.microsoft.com/sql
PostgreSQL	9.1 +	8.3.18 +	www.postgresql.org
Поддерживаемые веб-серверы:			
Apache (с mod_mysql, mod_xml, и mod_zlib)	2.4 +	2.x +	www.apache.org
Nginx	1.8 +	1.0 +	wiki.nginx.org
Microsoft IIS	7	7	www.iis.net

Перед установкой Joomla, необходимо провести подготовительные работы. Нужно скачать с официального сайта пакет с файлами для установки Joomla, затем необходимо создать базу данных. В данной системе будет использоваться две базы данных: одна – общая, в которой хранятся все данные, необходимые для работы системы, к которой будут обращаться, как и сайт, так и мобильные приложения; вторая база данных – это база, в которой будут хранится данные, необходимые для работы с сайтом. Связь с базой данных, необходимой для работы с сайтом, будет происходить через CMS Joomla. После того, как дистрибутив будет загружен, необходимо перенести файлы из папки upload в головную директорию веб-портала на сервере через FTP-клиент.

После загрузки фалов и создания базы данных необходимо перейти через браузер по адресу веб-портала, откроется веб-страница, показанная на рисунке 3.4:

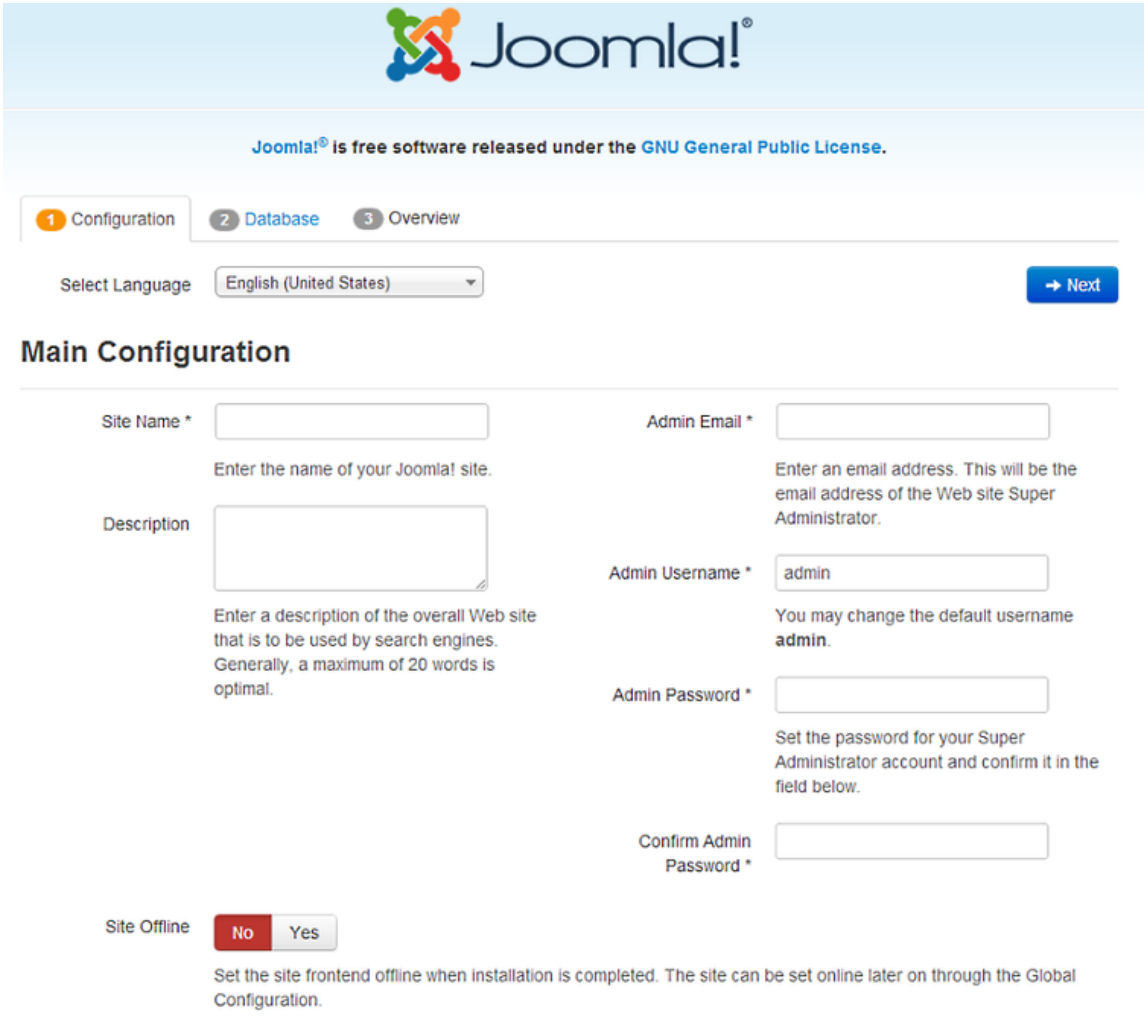
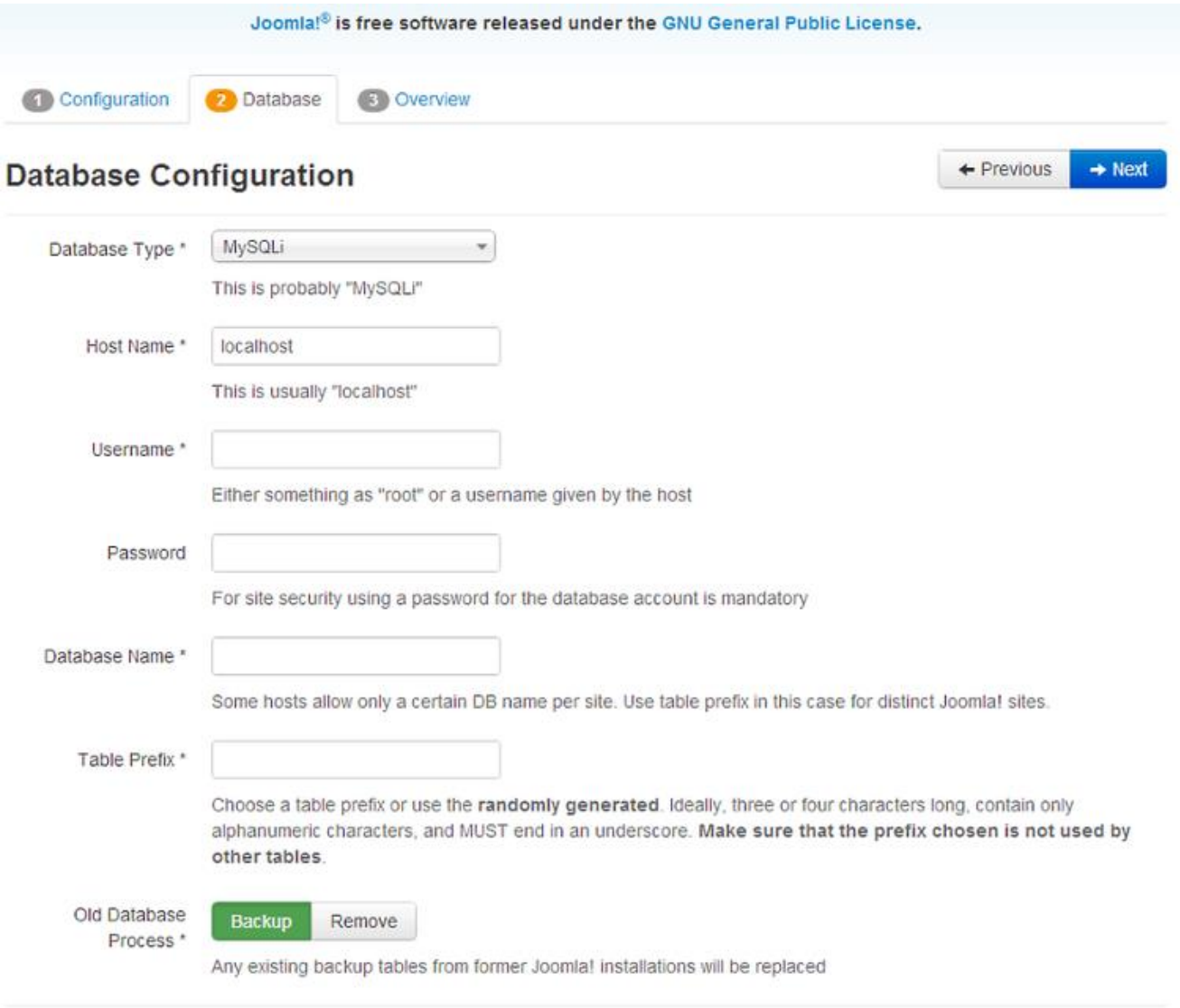


Рисунок 3.4 – Страница установки CMS Joomla

На данной странице необходимо указать данные администратора веб-портала. После этого нужно перейти на следующую страницу и указать данные, требуемые для доступа к базе данных. Страница показана на рисунке 3.5:



Страница 3.5 – Страница ввода данных для доступа к базе данных

На данной старнице вводятся следующие данные:

- Тип базы данных: в большинстве случаев используется MySQL, но я использую Postgres;
- Имя сервера базы данных: сервер, на котором хранится база данных;
- Имя пользователя: имя пользователя для соединения с этой базой данных;
- Пароль: пароль пользователя этой базы данных;
- Имя базы данных: название этой базы данных;

- Префикс таблицы: он обычно создается автоматически.

Действия с уже имеющимися таблицами: следует ли в процессе установки новых таблиц создать резервную копию новых таблиц или удалить их.

После окончания установки все эти варианты можно изменить на странице общих настроек сайта, в опциях параметра "Сервер". После ввода и подтверждения всех данных, установщик создает новые таблицы в базе данных и добавляет соответствующие записи. На этом установка CMS Joomla завершена.

3.4 Установка и настройка PostgreSQL

PostgreSQL используют такие компании, как Alibaba, Instagram, Skype, Yahoo и многие другие. Это говорит о надёжности системы, и при этом она является простой в установке, использовании и обслуживании.

PostgreSQL является кроссплатформенной СУБД с открытым исходным кодом, поэтому её можно установить практически на любой сервер.

Для установки базы данных необходимо добавить репозиторий PostgreSQL в системный список источников:

sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt/ `lsb_release -cs`-pgdg main" >> /etc/apt/sources.list.d/pgdg.list'

Во время установки программы в системе автоматически была создана учётная запись администратора баз данных — postgres. На данном этапе доступ к системе баз данных можно получить только через неё.

В СУБД после установки есть только одна роль — postgres. Данная роль не рекомендована для работы с пользовательскими базами данных, в целях безопасности необходимо создавать для каждой базы новую роль (или несколько при необходимости). Для создания новой роли предусмотрены два стандартных способа:

- интерактивный режим, в котором достаточно ответить на несколько простых вопросов;
- команда для создания роли через командную строку СУБД.

Для удобной работы с PostgreSQL также была установлена pgAdmin. pgAdmin – это бесплатная среда разработки и администрирования СУБД PostgreSQL. pgAdmin также, как и сама СУБД PostgreSQL реализована для всех известных платформ, таких как Linux, FreeBSD, Solaris, Mac OSX и Windows.

С помощью pgAdmin можно писать запросы, процедуры, функции, а также администрировать PostgreSQL, используя при этом отличный графический интерфейс. Интерфейс установленной pgAdmin показан на Рисунке 3.6:

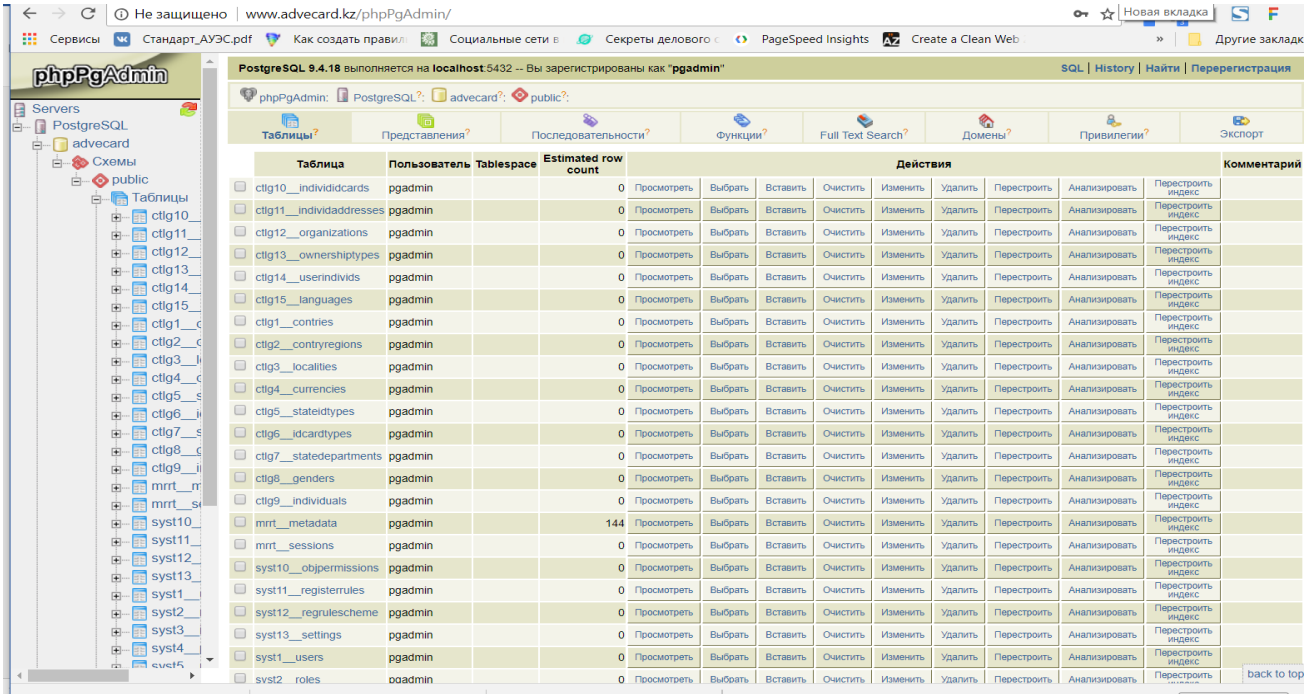


Рисунок 3.6 – Интерфейс pgAdmin

База данных проектировалась с таким учетом, чтобы в будущем её можно было легко масштабировать, добавлять новый функционал в систему и иметь легкий доступ к информации, хранящейся в таблицах. По данным из таблиц БД можно отследить того, кто внёс запись, время добавления записи, время изменения записи, количество изменений записи, можно оставлять комментарий к записи. Это очень удобно, когда записей становится очень много и необходимо отслеживать данные. Ниже перечислен список таблиц в созданной базе данных:

- ctlg10__individidcards
- ctlg11__individaddresses
- ctlg12__organizations
- ctlg13__ownershiptypes
- ctlg14__userindivids
- ctlg15__languages
- ctlg1__contries
- ctlg2__contryregions
- ctlg3__localities
- ctlg4__currencies
- ctlg5__stateidtypes
- ctlg6__idcardtypes
- ctlg7__statedepartments
- ctlg8__genders

- ctlg9__individuals
- mrrt__metadata
- mrrt__sessions
- syst10__objpermissions
- syst11__registerrules
- syst12__regrulescheme
- syst13__settings
- syst1__users
- syst2__roles
- syst3__interfaces
- syst4__points
- syst5__userroles
- syst6__userinterfaces
- syst7__pointusers
- syst8__interfacesettings
- syst9__reqpermissions
- utls__mtdtnumerator

База данных состоит из 34 таблиц, с первого взгляда многие подумают, что она громоздкая и некоторые таблицы можно было бы объединить, но в будущем, если они захотят расширить функционал системы, они поймут, что эти таблицы были созданы не зря. На Рисунке 3.7 показана диаграмма только основных таблиц в базе:

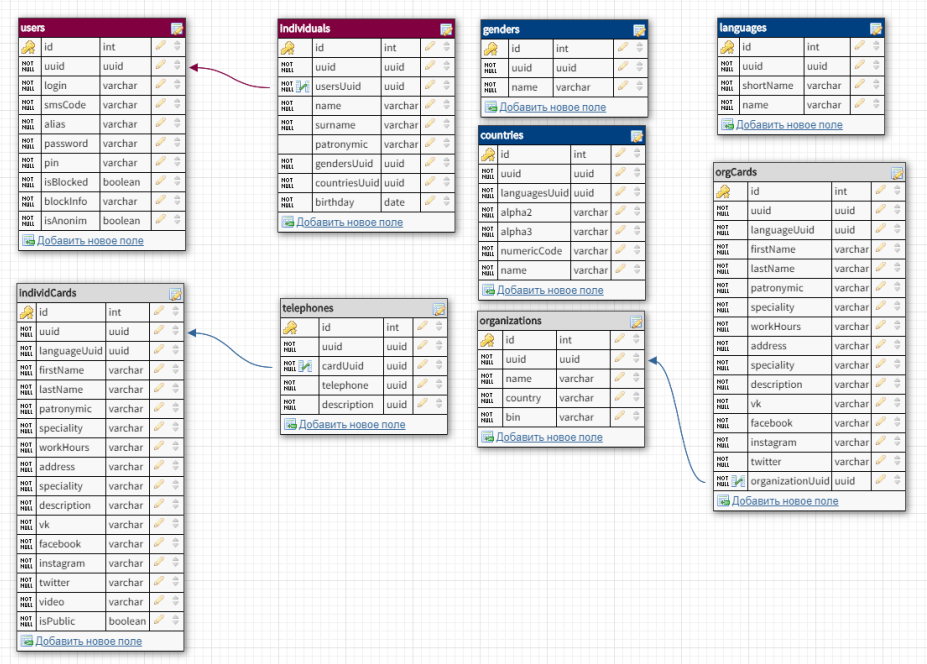


Рисунок 3.7 – Диаграмма основных таблиц в базе данных проекта

3.5 Интерфейс веб-портала

Весь функционал веб-системы построен на самописных компонентах и модулях, стандГлавная страница сайта сразу встречает посетителей списком последних добавленных Визиток. На десктопной версии сайта визитки расположены в два ряда, справа располагается главное меню. Главная страница показана на Рисунке 3.8:

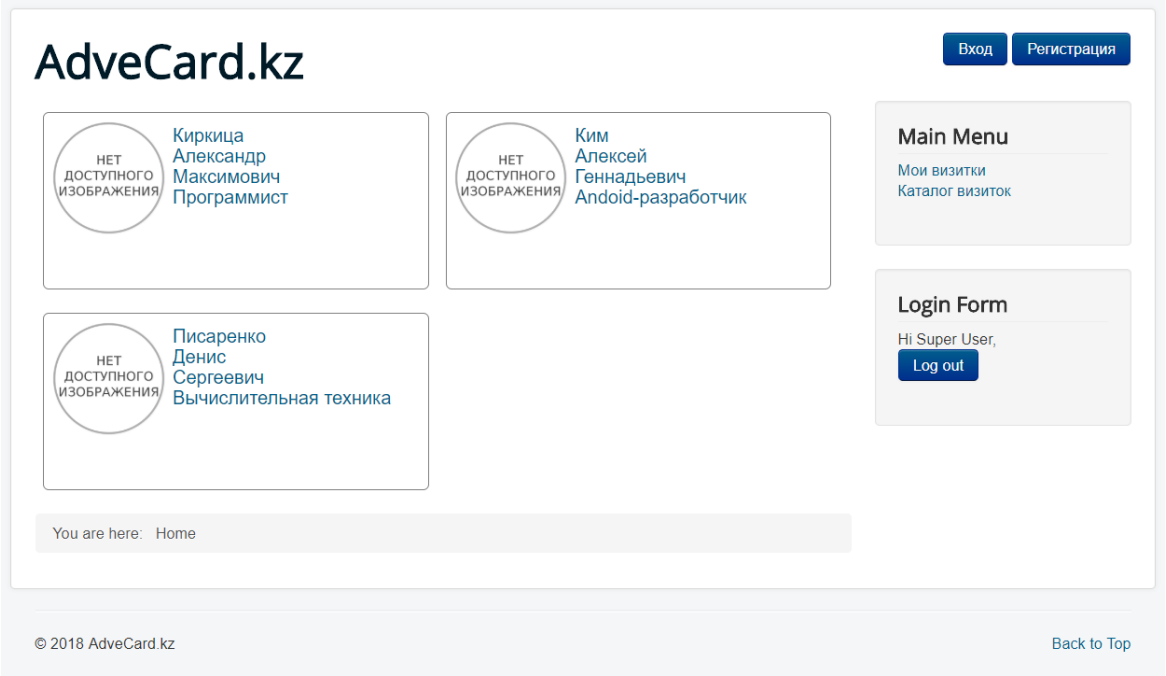


Рисунок 3.8 – Интерфейс главной страницы десктопной версии веб-портала

Все Визитки кликабельны, при нажатии на Визитку пользователь попадает на страницу с полным набором информации, которой обладает Визитка. В таком виде на визитках отображается только самая основная информация, такая как имя, фамилия, отчество и вид деятельности. Также отображается изображение, загруженное автором визитки. Для получения более подробной информации о человеке необходимо перейти на страницу отдельной визитки. Здесь уже отображается полное описание деятельности человека, его место работы и должность, территория, на которой пользователь предоставляет свои услуги. Автор визитки также может указать ссылки на свои личные страницы в социальных сетях. Ниже располагается блок с видеофайлами, загруженными на видеохостинг Vimeo. Изначально планировалось использовать в качестве видеохостинга YouTube, но в ходе разработки я пришел к решению, что YouTube не подходит для данного проекта. Загрузка видео на хостинг должна происходить через единый аккаунт системы AdveCard, но YouTube с 2016 года запрещает

загрузку видео без индивидуальной аутентификации. То есть, если бы я использовал YouTube, то каждый пользователь при загрузке видео должен бы был авторизоваться через окно Google OAuth 2.0 и видео бы загружалось на канал пользователя. Vimeo решает эту проблему. Был приобретен платный аккаунт Vimeo, который позволяет загружать видео пользователей от имени веб-приложения. Ниже, на Рисунке 3.9 показан интерфейс индивидуальной страницы визитки:

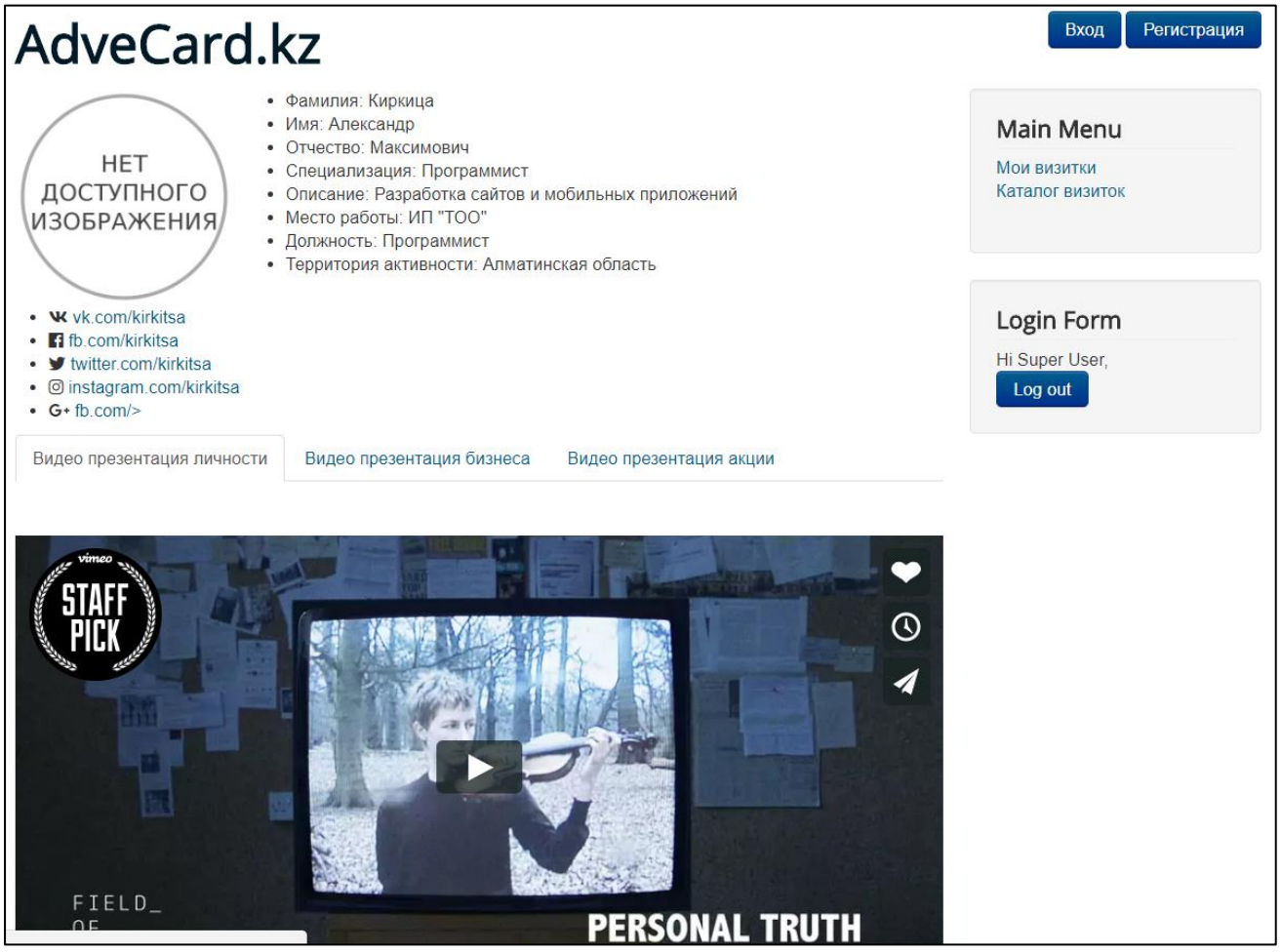


Рисунок 3.9 – Интерфейс индивидуальной страницы визитки

На странице «Мои визитки» отображаются все личные визитки пользователя. Если же пользователь только зарегистрировался и еще не создал ни одной визитки, у пользователя отображаются пустые окошки, куда можно добавить свои визитки. Предлагается три варианта визиток: публичные, приватные и фирменные. Контакты в публичных визитках видны всем пользователям системы, как авторизованным, так и анонимным. Приватные доступны только авторизованным пользователям и только с индивидуального

разрешения автора визитки. Фирменные визитки создаются от имени компании. Личные и фирменные визитки немного отличаются информацией, которую они могут содержать. На данный момент доступно создание одной визитки каждого вида на нескольких языках, в дальнейшем, в рамках монетизации появится возможность добавления дополнительных визиток. Страница «Мои визитки» показана ниже на Рисунке 3.10:

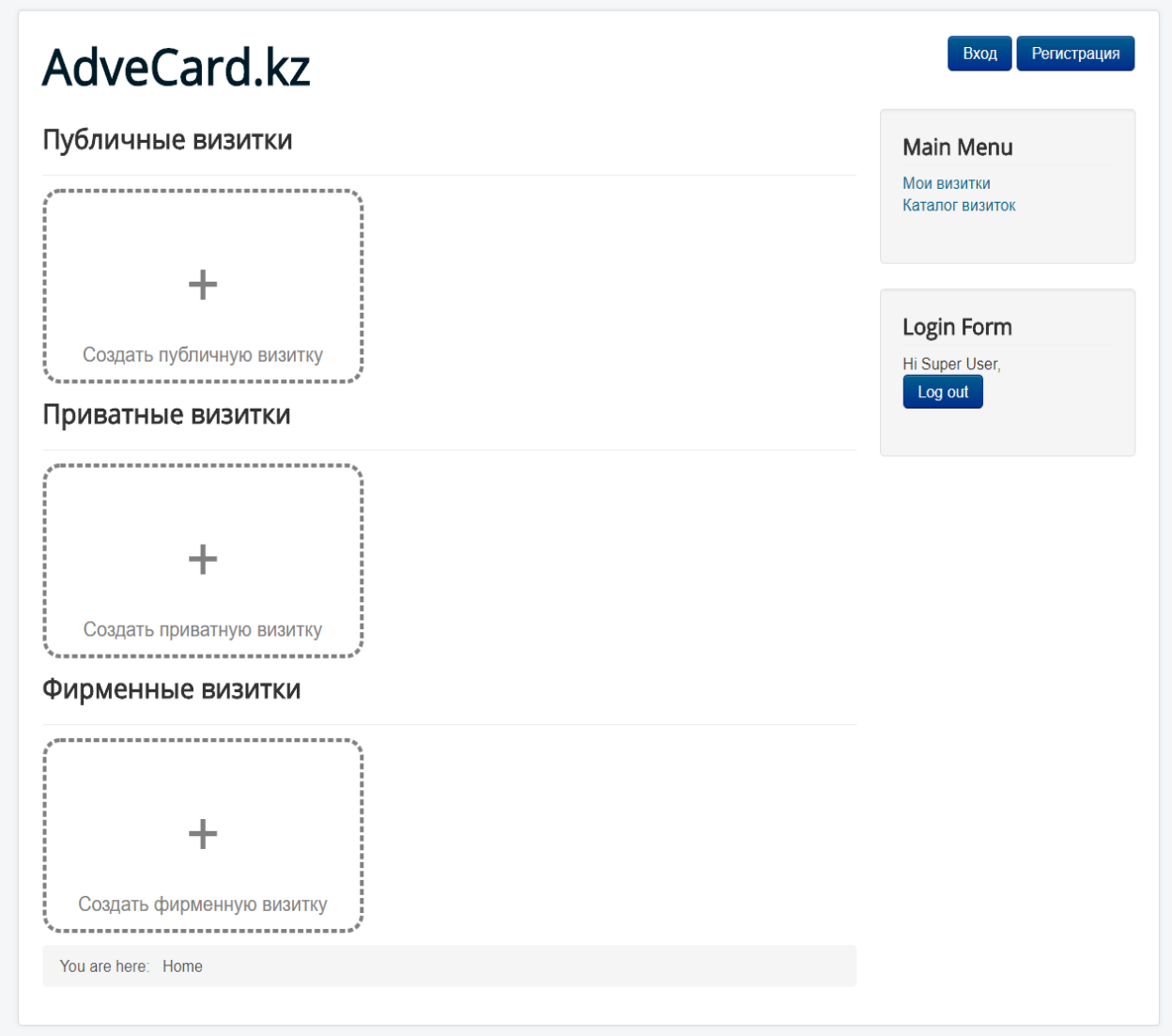


Рисунок 3.10 – Интерфейс страницы «Мои визитки»

При нажатии на кнопку создания визитки, всплывает модальное окно и пользователю предлагается выбрать язык, на котором она будет создаваться. Затем, при подтверждении выбора, в базу данных записывается выбранный язык, далее POST запросом язык передается на страницу создания визитки. Окно выбора языка показано на Рисунке 3.11:

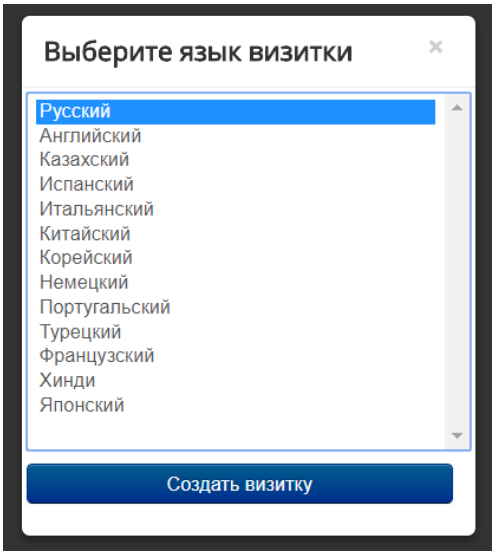


Рисунок 3.11 – Выбор языка создаваемой визитки

После выбора языка, пользователь попадает на пошаговую форму создания визитки. Первый шаг – ввод персональных данных пользователя, если пользователь создавал визитки раннее, то эти данные автоматически подгрузятся из БД и будут доступны для редактирования. Ввод персональных данных показа на рисунке 3.12:

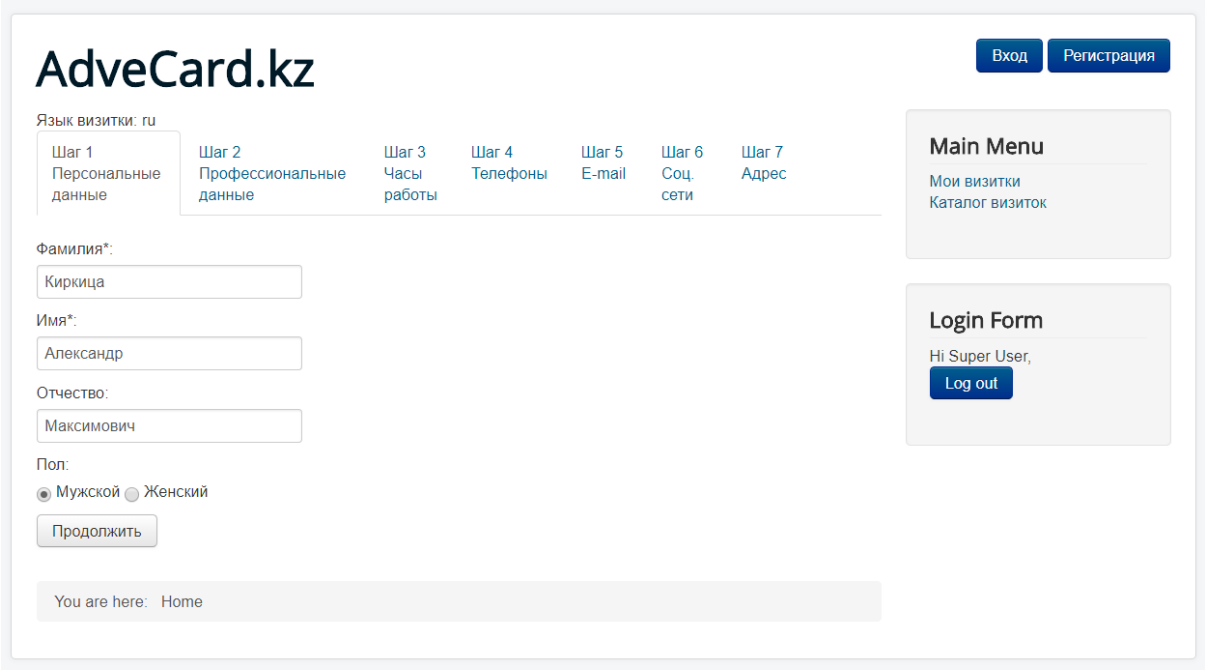


Рисунок 3.12 – Первый шаг создания Персональной Визитки «Ввод Персональных данных»

После ввода персональных данных необходимо заполнить данные о профессиональной деятельности пользователя. Затем пользователь выбирает часы работы в удобном интерфейсе, перемещая ползунки выбора диапазона рабочих часов для каждого дня недели. Данные шаг создания визитки показан на рисунке 3.13:

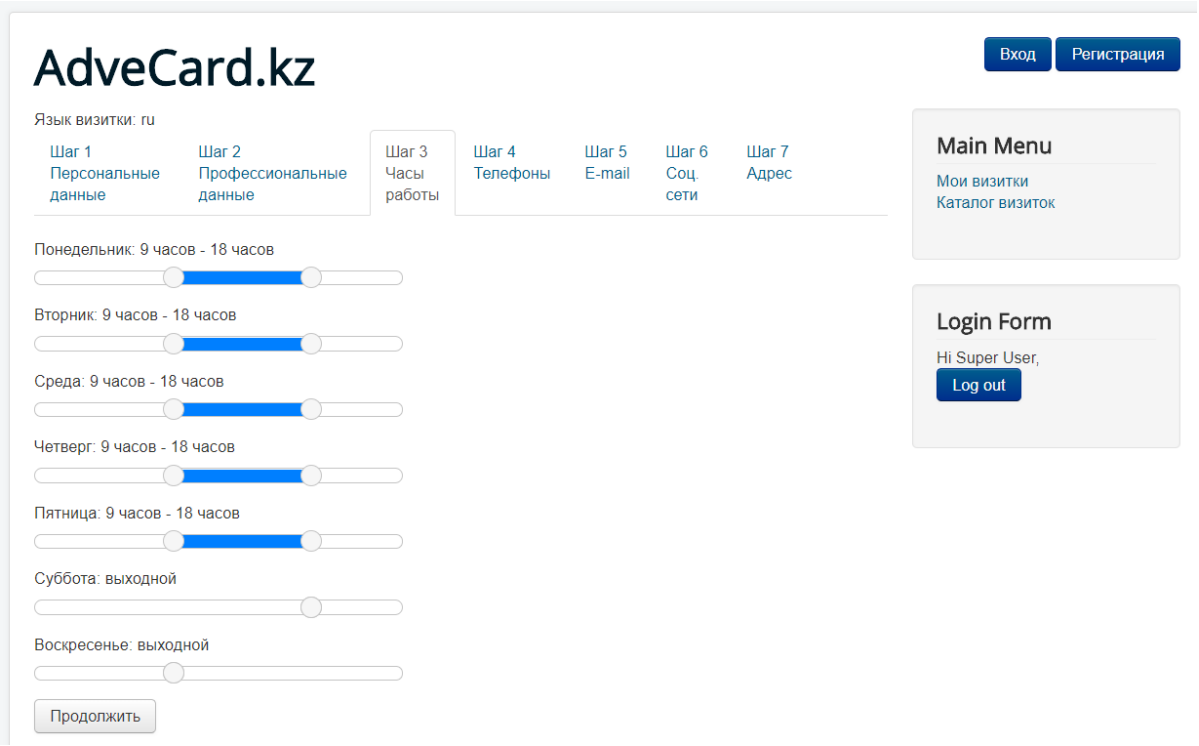


Рисунок 3.13 – Третий шаг создания Персональной Визитки «Часы работы»

На четвертом шаге пользователь должен указать номера своих телефонов, система предусматривает добавление до трех телефонов. Для каждого телефона можно указать наличие таких мессенджеров, как WhatsApp, Telegram и Viber. Также для каждого телефона создано текстовое поле, в которое пользователь может добавить дополнительную информацию. На пятом шаге вводятся e-mail адреса пользователя, также с примечанием для каждой почты. Шестой шаг – ввод ссылок на личные страницы пользователя в социальных сетях. Седьмой шаг – указание территории активности. Территория активности – это та территория, в пределах которой пользователь оказывает свои услуги. Выбор территории активности происходит на карте Google Maps. При выборе точки на карте, ситема отправляет координаты точки в сервис Google Maps и возвращает название страны, области и населенного пункта. Пользователь выбирает из этих трех значений территорию, на которой он работает. Выбор территории активности показан на рисунке 3.14:

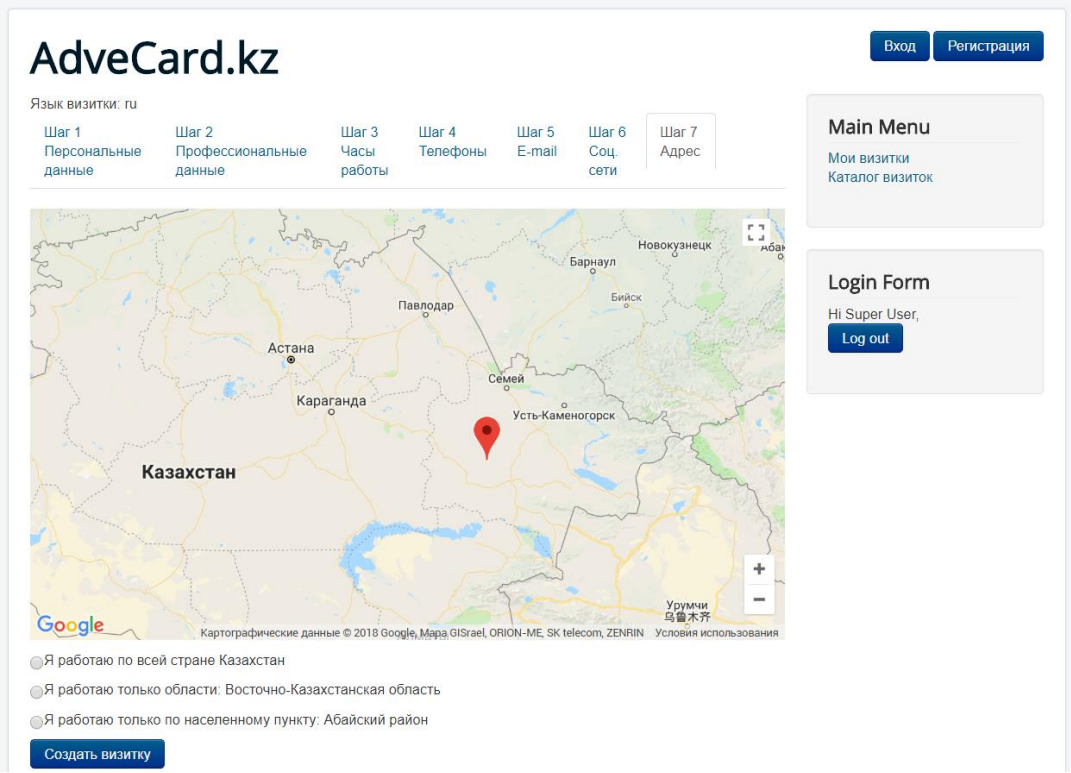


Рисунок 3.14 – Седьмой шаг создания Персональной Визитки «Ввод адреса»

На этом создание визитки закончено. При редактировании визитки можно будет добавить дополнительную информацию, например, видео-презентацию. Ввод дополнительной информации показан на рисунке 3.15:

Рисунок 3.15 – Страница редактирования Персональной Визитки «Ввод доп. информации»

Регистрация в системе начинается с ввода номера телефона, система автоматически, на основе маски номера телефона, определяет страну, в которой зарегистрирован телефон. После соглашения с условиями Договора Оферты кнопка «Продолжить» становится активной. Если нажать на эту кнопку, на введённый номер отправляется СМС с кодом подтверждения номера телефона через API сервиса smsc.kz. Модальное окно ввода номера телефона показано на рисунке 3.16:

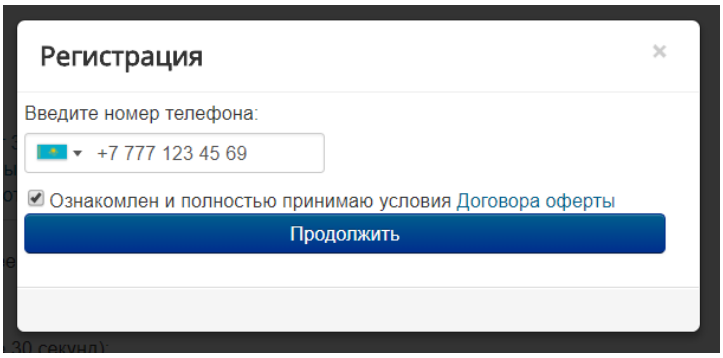


Рисунок 3.16 – Модуль регистрации в системе Шаг 1

После отправки СМС открывается окно, в котором необходимо ввести код, полученный в СМС и пароль. Минимальная длина пароля составляет 6 символов. После нажатия кнопки «Зарегистрироваться» пользователь попадает на страницу «Мои визитки». На рисунке 3.17 показан второй шаг регистрации:

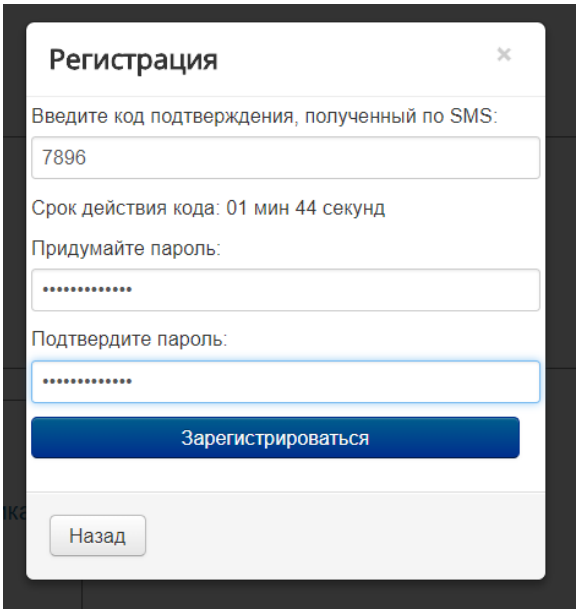


Рисунок 3.17 – Модуль регистрации в системе Шаг 2

Если пользователь уже зарегистрировался в системе, то авторизоваться он может введя номер своего телефона и пароль. Вход в систему показан на рисунке 3.18:

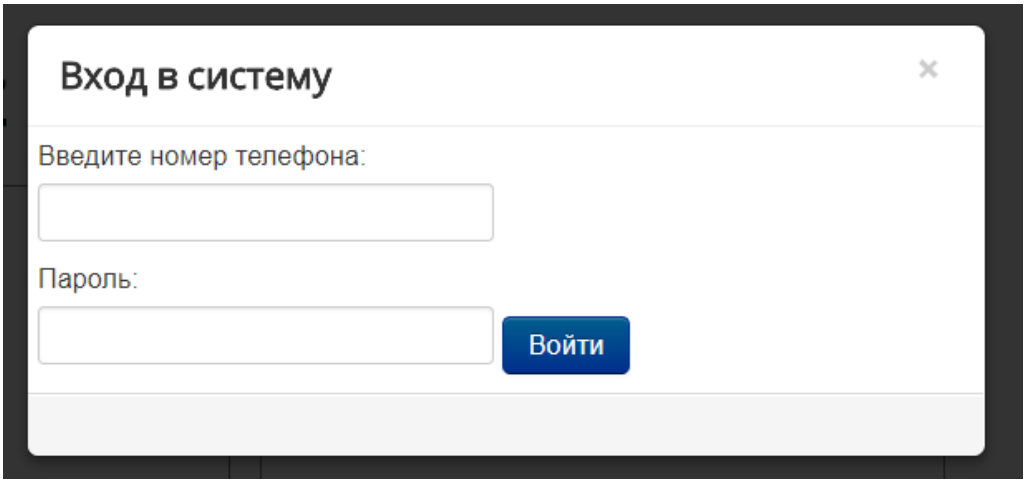


Рисунок 3.18 – Модуль входа в систему

На рисунке 3.19 изображена диаграмма вариантов использования:

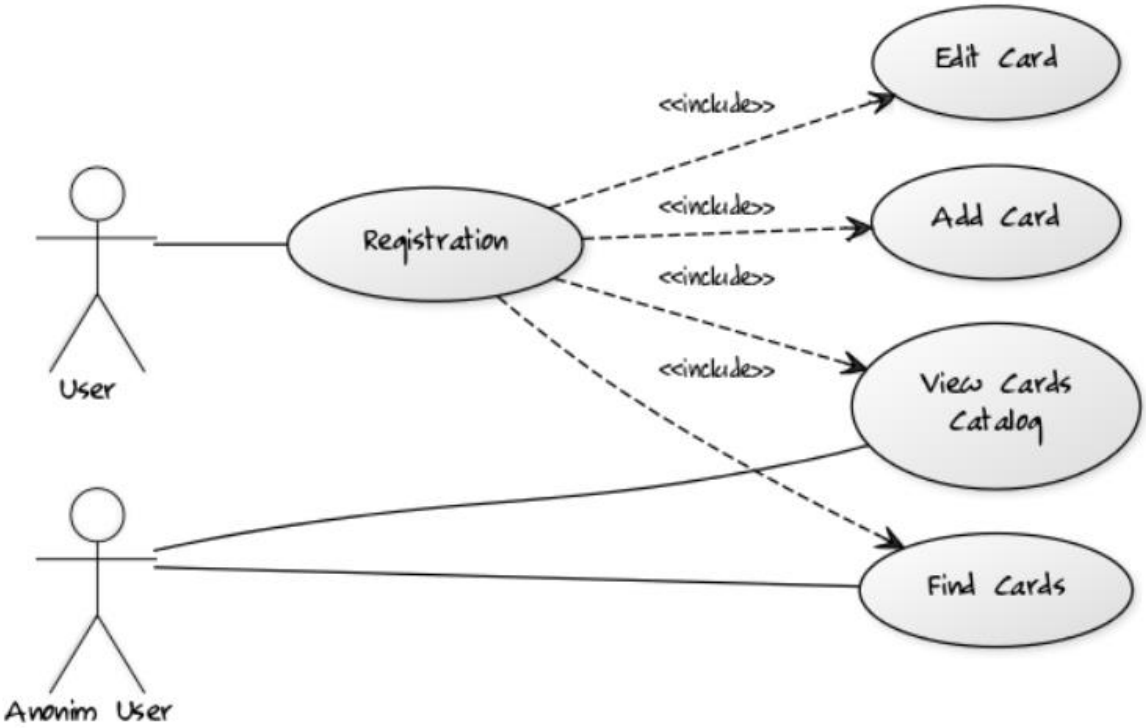


Рисунок 3.19 – Диаграмма вариантов использования

4 Технико-экономическое обоснование

4.1 Введение технико-экономического обоснования

Данный раздел дипломного проекта рассматривает все затраты, необходимые для полноценного функционирования программного продукта.

Тема данного дипломного проекта – «Технологии разработки web-приложений (на примере компании «Web Line»)».

Данный проект призвана усовершенствовать процесс обмена деловой информацией, актуализирует контакты, экономит время и деньги на создание визиток, а также полезна с экологической точки зрения, так как экономит бумагу.

Целью данного дипломного проекта является создание веб-портала, предоставляющего замену бумажному носителю контактной информации о человеке или организации на электронную версию, при этом расширив представительские возможности визиток с помощью видео, аудио, графического контента, интерактивных карт и пр.

4.2 Расчет трудоемкости разработки ПО.

Расчет плановой сметы затрат на разработку программного продукта производится исходя из его объема. Общий объем (V_0) программного обеспечения вычисляется исходя из количества и объема функций, составляющих данное программное обеспечение показан в Формуле 4.1:

$$V_0 = \sum_{i=1}^{n} V_i, \quad (4.1)$$

где Vi − объем отдельной функции ПО;

n − общее число функций.

Объем программного обеспечения, рассмотренного в данном дипломном проекте равен 16000 (строки исходного кода, LOC). Принимаем: V_0 = 16000.

Общая трудоемкость процесса рассчитывается по Формуле 4.2:

$$T_0 = К_с * К_т * К_m * К_н, \quad (4.2)$$

где $К_с$ − коэффициент, учитывающий сложность ПО;

$К_т$ − поправочный коэффициент, учитывающий степень использования при разработке стандартных модулей;

$К_н$ − коэффициент, учитывающий степень новизны ПО.

Коэффициент сложности базы данных и приложения определяется на основе дополнительных коэффициентов сложности ПО из Таблицы 4.1 и равен

0,12, так как данное ПО имеет две следующие характеристики: функционирование ПО в расширенной операционной среде (связь с другими ПО) и интерактивный доступ.

Таблица 4.1 – Дополнительные коэффициенты сложности ПО

Характеристика ПО	Значения $К_с$
1. Функционирование ПО в расширенной операционной среде (связь с другими ПО)	0,08
2. Интерактивный доступ	0,06
3. Обеспечение хранения, ведения и поиска данных в сложных структурах	0,07
4. Наличие у ПО одновременно нескольких характеристик по табл. 4.1 4.1 2 характеристики 4.2 3 характеристики 4.3 Свыше 3-х характеристик	 0,12 0,18 0,26

Поправочный коэффициент, учитывающий степень использования при разработке ПО основных модулей ($К_т$), определяется исходя из данных таблицы 4.2 и равен 0,8.

Таблица 4.2 – Значения поправочного коэффициента, учитывающего использование стандартных модулей типовых программ и ПО ($К_т$)

Степень охвата реализуемых функций разрабатываемого ПО стандартными модулями, типовыми программами и ПО	Значения $К_т$
1. От 60 % и выше	0,6
2. От 40 % до 60	0,7
3. От 20 % до 40 %	0,8
4. До 20 %	0,9
5. Типовые программы и ПО, не используемые для реализации функций разрабатываемого ПО	1,0

Поправочный коэффициент, учитывающий новизну разрабатываемого ПО ($К_н$) определяется на основе данных таблицы 4.3 и равен 1,0.

Входными данными для расчета нормативной трудоемкости при разработке данного ПО являются заявленные нормы времени на разработку проекта в зависимости от точечного объема программного обеспечения и соответствующей ему группе сложности.

Таблица 4.3 – Поправочные коэффициенты, учитывающие новизну ПО ($К_н$)

Категория новизны	Степень новизны	Использование		Значение $К_н$
		На основе нового типа ПК	В среде новой ОС	
		+	+	1,75
А	Принципиально новые ПО, не имеющие доступных аналогов	– + –	+ – –	1,6 1,2 1,0
Б	ПО, являющиеся развитием определенного параметрического ряда ПО	+ – +	+ – –	1,0 0,9 0,8
В	ПО, являющиеся развитием определенного параметрического ряда ПО, разработанных для ранее освоенных типов конфигурации ПК и ОС	–	–	0,7

Нормативная трудоемкость проекта ($Т_н$) в свою очередь определяется на основе принятого в расчет объема программного продукта и категории сложности, которая уточняется с учетом сложности и новизны проекта и степени использования стандартных модулей при разработке.

Учитывая исходные данные для данного проекта: для 1–ой категории сложности ПО $Т_н$ =406.

Рассчитаем общий объем трудоемкости по формуле (4.2):

$$Т_0 = 406 * 0{,}12 * 0{,}8 * 1 = 38{,}98$$

На основе трудоемкости определяются плановое число разработчиков ($Ч_р$) и плановые сроки, необходимые для проекта($Т_р$). При этом могут решаться нижеперечисленные задачи:

- расчет числа исполнителей при заданных сроках разработки проекта;

- определение сроков разработки проекта при заданной численности исполнителей.

Численность исполнителей проекта ($Ч_р$) рассчитывается по Формуле 4.3:

$$Ч_р=Т_0/(Т_р*Ф_{эф}), \quad (4.3)$$

где $Ф_{эф}$ − эффективный фонд времени работы одного работника в течении года (дн.);

$К_0$ − общая трудоемкость разработки проекта (чел./дн.);

$Т_р$ − срок разработки проекта (лет).

Срок разработки базы данных и приложения ($Т_р$) определяется по Формуле 4.4:

$$Т_р=Т_0/(Ч_р*Ф_{эф}), \quad (4.4)$$

где $Ч_р$ – плановое число разработчиков.

Эффективное время работы одного работника ($Ф_{эф}$) рассчитывается по данной Формуле 4.5:

$$Ф_{эф}=Д_г+Д_п+Д_в+Д_о, \quad (4.5)$$

где $Д_г$ − количество дней в году;

$Д_п$ − количество праздничных дней в году;

$Д_в$ − количество выходных дней в году;

$Д_о$ − количество дней отпуска.

Сопоставив данные с производственным календарем на 2018 год.: $Д_г$ − 365; $Д_п$ = 16; $Д_в$ – 103; $Д_о$ – 15, эффективный фонд времени одного программиста должно составить:

$$Ф_{эф}=365−16−103−15=231 \text{ дня}$$

Плановое число разработчиков программного обеспечения $Ч_р$ = 1, следовательно, по формуле (2.4)

$$Т_р=38,98/1*231=0.17 \text{ лет} = 62.05 \text{ дня}$$

Таким образом, учитывая произведенные расчеты и формулу (2.3) имеем следующие данные:

$$Ч_р=38,98/0.17*231=1 \text{ чел}$$

4.3 Расчет затрат на разработку веб-портала

Расчет полных затрат на разработку проектного решения в виде веб-портала ($С_{пі}$) осуществляется по Формуле 4.6:

$$С_{пі} = З_{фот} + З_{сзі} + М_і + Р_{сі} + Р_{мі} + Р_{нкі} + П_{зі} + Р_{ні}, \quad (4.6)$$

где $З_{фот}$ – общий фонд оплаты труда разработчиков, тенге;

$З_{сзі}$ – отчисления по социальному налогу, тенге;

$М_і$ – затраты на материалы, тенге;

$Р_{сі}$ – затраты на специальные программные средства, необходимые для разработки проектного решения, тенге;

$Р_{мі}$ – затраты, связанные с эксплуатацией техники, тенге;

$П_{зі}$ – прочие затраты, тенге;

$Р_{ні}$ – накладные расходы, тенге.

Размер фонда оплаты труда разработчиков ($З_{ФОТ}$) рассчитывается по формуле:

$$З_{ФОТ} = З_о + З_д, \quad (4.7)$$

где $З_о$ – основная заработная плата, тенге;

$З_{д.}$ – дополнительная заработная плата, тенге.

Основная заработная плата исполнителей на создание веб-портала рассчитывается по формуле:

$$З_о = \sum_{i=1}^{n} Т_{чi} * Т_ч * Ф_n * К, \quad (4.8)$$

где n − количество исполнителей, занятых разработкой конкретного ПО;

$Т_{чі}$ − часовая тарифная ставка і–го исполнителя (тыс. тенге);

$Ф_п$ − плановый фонд рабочего времени і–го исполнителя (дней), 21 раб день в месяц;

$Т_ч$ − количество часов работы в день (час), 8 часов;

К − коэффициент премирования, составляет 1,5.

По данным о специфике и сложности выполняемых функций составляется штатное расписание группы специалистов–исполнителей, участвующих в разработке ПО, с определением образования, специальности, квалификации и должности (таблица 4.4).

Часовая тарифная ставка рассчитывается путем деления месячной тарифной ставки, установленную при 40–часовой недельной норме рабочего времени и общего фонда времени($Ф_р$)

$$Т_ч=Т_м/Ф_р, \quad (4.9)$$

где $Т_ч$ – часовая тарифная ставка (тыс. тенге);
$Т_м$ – месячная тарифная ставка (тыс. тенге).

Таблица 4.4 – Сведения по работникам, задействованным в проекте

Специалист - Исполнитель	Количество, человек	Заработная плата в месяц, тенге
IT специалист	1	140000
Итого	1	140000

Общий фонд времени:

$$Ф_р=Т_ч*Ф_п, \quad (4.10)$$

Таким образом:

$$Ф_р=8*21=168 \text{ ч.}$$

Рассчитаем тарифную ставку инженера-программиста:

$$Т_ч =140\ 000/168=833 \text{ тг. в час}$$

Основываясь формулой (4.8) основная заработная плата инженера-программиста соответственно следующая:

$$З_о=833*8*21*1.2=167933 \text{ тг.}$$

Дополнительная заработная плата составляет 10% от основной заработной платы и рассчитывается по следующей формуле:

$$З_д=З_о*Н_д/100, \quad (4.11)$$

где $Н_д$ – коэффициент дополнительной заработной платы разработчиков 23%.

$$З_д=167933*0,23=38625$$

Социальный налог составляет 11% (ст. 358 п. 1 НК РК) от дохода работника, и рассчитывается по формуле:

$$З_с=(ФОТ-П)*11\%, \quad (4.12)$$

где П – пенсионные отчисления, которые составляют 10% от ФОТ и социальным налогом не облагаются

$$П=ФОТ*10\%, \quad (4.13)$$

Таким образом:

$$П=167933*0,1=16793 \text{ тг.}$$
$$З_с=(167933-16793)*0,11=16625 \text{ тг.}$$

Затраты на материалы определяются по формуле:

$$М_i=(З_о*Нмз)100\%, \quad (4.14)$$

где $Н_{мз}$ – норма расхода материалов от основной заработной платы (3–5%).

$$М_i=167933*0,045=7557 \text{ тг.}$$

Расходы по статье «Спецоборудование» ($Р_с$) составляют 218 750 тенге без НДС, с учетом НДС $Р_с$ = 245 000 тенге. Сводные затраты по стоимости оборудования представлены в таблице 4.5

Таблица 4.5 – перечень оборудования, используемого при реализации проекта

Оборудование	Марка, характеристики	Кол-во единиц	Стоимость без НДС, тг	Стоимость с НДС, тг	Общая сумма, тг
Ноутбук	ASUS N550JK, Intel Core i7-4710HQ, 2,50Ghz, RAM 8gb, ROM 1tb.	1	218750	245000	245000
Итого					245000

Расходы по части «Машинное время» $Р_м$ включают оплату машинного времени, который необходимо для реализации и отладки программы, определяющийся по нормативам (в машино – часах) на 100 строк исходного кода ($Н_{мв}$) машинного времени в зависимости от характера решаемых задач и типа персонального компьютера:

$$Р_м = Ц_м \cdot (V_о / 100) \cdot Н_{мв}, \quad (4.15)$$

где $Ц_м$ − цена одного машинного часа (тыс.тенге);

$Н_{мв}$ − норматив расхода машинного времени на отладку 100 строк исходного кода (машинно–часов);

$V_о$ − общий объем ПО (строк исходного кода).

Норматив расхода машинного времени на отладку 100 строк исходника составляет 12 ч/100 строк кода

$$Р_м = 0.833 \cdot (16000/ 100) \cdot 12 = 1599.360 \text{ (тенге)}$$

Расходы по статье «Прочие затраты» ($П_з$) на конкретное продукт, либо сервис включают затраты на приобретение и подготовку специальной научно–технической информации и специальной литературы. Определяются по нормативу, разрабатываемому в целом по организации, в процентах к основной заработной плате:

$$П_з=З_о*Н_{пз}/100, \quad (4.16)$$

где $Н_{пз}$ − норматив прочих затрат в целом по организации (20%)

Таким образом:

$$П_з=167933*0,2=33587тг.$$

Затраты по статье «Накладные расходы» ($Р_н$), рассчитывается по нормативу ($Н_{рн}$) в процентном отношении к основной заработной плате исполнителей. Норматив устанавливается в целом по организации:

$$Р_н=З_о* Н_{рн} *100\%, \quad (4.17)$$

где $Р_н$ – накладные расходы на конкретную ПО (тыс.тенге);

$Н_{рн}$ – норматив накладных расходов в целом по организации (70%).

Таким образом накладные расходы составят:

$Р_н$=167933∗0,7=117553тг.

В соответствие с формулой (4.6) полные затраты на разработку веб-портала, составит:

Сп*i*=167933+16625+7557+245 000+ 1599.36+33587+117553=590054.36тг

Таблица 4.6 – Затраты на разработку

Затраты на разработку	Условное обозначение	Значение, тенге	В процентах от общей суммы
Фонд оплаты труда	$З_{ФОТ}$	167933	28,46 %
Социальный налог	$З_{сзі}$	16625	2,82%
Материалы	$М_і$	7557	1,28%
Спецоборудование	$Р_{сі}$	245 000	41,52%
Машинное время	Рмі	1 599.36	0,27%
Прочие затраты	$П_{зі}$	33587	5,69%
Накладные расходы	$Р_{ні}$	117553	19,92%
Итого:		590054.36	100,0%

Рисунок 4.1 – Структура затрат на разработке веб-портала

4.4 Расчет цены программного продукта

Расчет цены ПП, который разработан одной компанией по заказу другой и не предназначен для тиражирования, осуществляется по Формуле 4.18:

$$Ц_{пп} = З_{рпр} + П_{п} + НДС, \quad (4.18)$$

где $Ц_{пп}$ – цена веб-портала, тенге;

$З_{рпр}$ – затраты на разработку проектного решения, в данном случае программного продукта, тенге;

$П_{п}$ – планируемая прибыль, тенге;

НДС – налог на добавленную стоимость, тенге.

Планируемая прибыль составляет (20%) от себестоимости разработки.

$$П_{п} = 590054.36 * 0,2 = 119810 \text{ тг.}$$

НДС, начисленный на ПП, определяется следующим образом:

$$НДС = (З_{рпр} + П_{п}) * k НДС, \quad (4.19)$$

где kНДС – ставка налога на добавленную стоимость.

Подставив данные в формуле (4.19) получаем:

$$НДС = (590054.36 + 119810) * 0.12 = 85184 \text{ тг.}$$

Подставив данные в формуле (4.18) получаем:

$$Ц_{пп} = 590054.36 + 119810 + 85184 = 795048.36 \text{ тг}$$

4.5 Вывод по технико – экономической части

Создание веб-портала такого формата является действительно актуальным и востребованным, одним из главных преимуществ является то, что данное ПО упростит обмен деловой информацией между людьми. В технико–экономической части данного дипломного проекта получены данные по всем издержкам и расходам на разработку данного программного продукта. Себестоимость разработки составляет 590054.36 тенге. Планируемая цена реализации составляет 795048.36 с учётом НДС.

Внедрение данного программного обеспечения усовершенствует обмен деловой информацией, сделает эту информацию более интерактивной (в такой визитке, в отличие от бумажной есть возможность хранения видео, фотографий, ссылок на социальные сети), вся информация будет всегда актуальной (начиная от графика и места работы, заканчивая проводимыми на данный момент акциями). Бумажные визитки часто теряются, для того чтобы не потерять электронные визитки – достаточно помнить номер вашего телефона. Используемые технологии позволяют работать данной системе не только как сайту, но также и как приложению для операционных систем Android и IOS.

5 Безопасность жизнедеятельности

5.1 Введение по разделу безопасности жизнедеятельности

Данная дипломная работа призвана усовершенствовать процесс обмена деловой информацией, позволяет быть вашим контактам всегда актуальными, экономит время и деньги на создание визитки, а также несет пользу с экологической точки зрения, ведь теперь для визиток не нужна бумага.

Визитная карточка – это карточка, на которой указана основная контактная информация человека. Бумажные визитки очень ограничены в количестве информации, которую они могут передать. Для того, чтобы создать визитку, необходимо большое количество времени на разработку дизайна и поход в типографию. Электронную же визитку можно создать за 5 минут: для этого нужно просто зайти на сайт и зарегистрироваться. Большим преимуществом электронных визитных карточек является то, что они не заканчиваются, и если у вас меняется какая-то контактная информация, то вы просто редактируете её в своем профиле и она автоматически меняется у всех обладателей вашей визитки.

Данное программное обеспечение будет использоваться на деловых встречах, в конференц-залах.

В данном разделе дипломной работы будет рассмотрены понятия вентиляции, рассчитана вентиляция конференц-зала.

5.2 Анализ условий труда в помещении конференц-зала

Помещение – конференц-зал S = 160 м^2, вместимостью до 120 человек. В помещении имеется два окна с ориентацией на СВ и два окна с ориентацией на СЗ.

Площадь окон:

$$F_{\text{окн}} = (1{,}47 \cdot 1{,}76) \cdot 6 = 15{,}52\text{м}^2 \quad (5.1)$$

Для создания благоприятных условий микроклимата необходимо, чтобы температура внутренних поверхностей ограждений помещений (стены, перекрытия) не была ниже температуры воздуха более чем на 4—6°. При большей разности температур возрастают потери тепла излучением во внешнюю среду и возникает ощущение зябкости. В определении зоны комфорта следует учитывать воздействие на организм лучистого тепла (прямые солнечные лучи, нагретые поверхности отопительных приборов, ограждений помещений и т. д.). В этих условиях температурные границы зоны комфорта снижаются.

Вентиляция конференц-залов отличается дискретностью и периодичностью своей работы. Она работает максимально интенсивно во время проведения мероприятий, когда человеческая нагрузка усиливается, а в остальное время использует минимум мощности или выключена вовсе, в зависимости от характера использования помещения. Система кондиционирования конференц-залов должна быть рассчитана на большие площади, а действие равномерно распределено по периметру, чтобы не вызывать дискомфорта у сидящих в непосредственной близости людей.

В залах со сценой воздухообмен определяется по расчету, но не менее 20 $м^3$/ч наружного воздуха на 1 зрителя.

В помещениях создается подпор в размере 20% по балансу воздухообмена. Основная проблема при расчете и конструировании систем вентиляции и кондиционирования это — загрузка зала. По статистике полная загрузка зала встречается в 10% случаев, расчетные метеорологические условия также происходят не чаще.

Основное назначение приточно-вытяжных систем вентиляции с рекуперацией тепла – это, во-первых, решение проблем с организацией подачи свежего приточного воздуха в помещения и удалением отработанного вытяжного воздуха из помещения. Во-вторых, экономия тепловой или электрической энергии на нагрев холодного приточного воздуха благодаря специальному утилизатору теплоты, который забирает тепло из удаляемого воздуха и передает его свежему подаваемому в помещение воздуху. Другими словами, воздух, который удаляется, обычно имеет температуру от 20 до 30 °С. Его тепло и используется для подогрева холодного воздуха в специальном теплообменнике – рекуператоре. В холодный и переходный период года такие системы, в зависимости от типа рекуператора, позволяют экономить до 85 % энергии необходимой на нагрев наружного приточного воздуха.

Как правило, в состав приточно-вытяжных вентиляционных установок с рекуперацией тепла входят два вентилятора – приточный и вытяжной, рекуператор (может быть пластинчатый или роторный), два фильтра – для приточного и вытяжного воздуха, нагреватель (водяной или электрический), а также система автоматики с пультом управления.

В пластинчатых рекуператорах приточный и вытяжной потоки воздуха полностью разделены, системой автоматики предусмотрена защита от обмерзания теплообменника. Эффективность таких установок может достигать 60-70 %. В агрегате с роторным теплообменником тепловая энергия уходящего воздуха накапливается в ячейках вращающегося рекуператора барабанного типа и передается входящему холодному потоку воздуха, нагревая его. В данном случае происходит небольшое попадание вытяжного воздуха к приточному,

порядка 5-10 %. Процент возврата тепла может достигать значения 85%. Роторные рекуператоры более эффективны с точки зрения энергосбережения.

Оборудование вентиляционных приточно-вытяжных систем с рекуперацией тепла, достаточно компактно и занимает не много места при его монтаже. Как правило, такие агрегаты имеют моноблочную конструкцию, поэтому их транспортировка и установка на объекте не вызывает трудности.

Теплоизбытки, которые не снял приточный воздух, снимаем за счет установки внутренних блоков системы кондиционирования фирмы "LG".

Самое большое количество теплоизбытков, исходя из предыдущих расчетов, в переходный период года равняется 44674 Вт, а объемные расходы воздуха по притоку и вытяжке $L_{\text{П}}^{\text{Р}} = 12491 \text{м}^3/\text{ч}$ и $L_{\text{У}}^{\text{Р}} = 12806 \text{м}^3/$ чсоответственно.

Рисунок 5.1 – Внутренний и наружный блоки кондиционера

Таблица 5.1 - Подбор внутренних блоков кондиционера

Qт.изб., кВт	Холодопроизводительность внутреннего блока $Q_{вн}$, кВт	Тип внутреннего блока	Марка внутреннего блока	Расход воздуха, $м^3$/мин	Количество, шт.
18	5,6	настенный	FDTC60 VF	2,0	12

По каталогу фирмы "LG" подбираем наружный блок LG FM57AH мультизональной системы кодиционирования и выписываем его технические характеристики:

Таблица 5.2 - Технические характеристики наружного блока 4MXS80E

Габариты (ШхВхГ), см	950*1380*330
Масса, кг	59
Хладагент	R410A
Диапазон рабочих температур: Охлаждение Нагрев	 -10~43 -10~24
Электропитание (VM)	3/380-415/50 В,Гц

5.3 Расчет и подбор воздухораспределительных устройств

Диффузоры предназначены для применения в вытяжных системах вентиляции и кондиционирования. Они представляют собой потолочные воздухораспределительные элементы с плавным регулированием расхода воздуха, которое осуществляется с помощью вращения центрального диска. Диффузоры изготавливаются из стали и имеют защитное порошковое покрытие белого цвета. Для удобства монтажа диффузоры снабжены соединительной муфтой, с помощью которой они присоединяются к воздуховодам.

Для расчета количества диффузоров используют следующую формулу:

$$N = \frac{L}{(2820 \cdot V \cdot d^2)}; \quad (5.2)$$

где N – количество диффузоров;

L – расход воздуха, 1600 м^3/час;

V – скорость движения воздуха на диффузоре, м/сек;

d – диаметр диффузора (принимается 0,2), м;

Скорость движения воздуха в приточных и вытяжных решетках принимаем не более 2,5 м/с.

$$N = \frac{1600}{(2820 \cdot 1.2 \cdot 0.2^2)} = 11{,}8$$

Т. о. необходимо не более 12 диффузоров.

В качестве воздухораспределительных устройств выбираем потолочные квадратные диффузоры марки DLRH 400-6 размером 565х569 мм, предназначенные для применения в приточных и вытяжных системах вентиляции и кондиционирования воздуха.

Диффузоры DLRH состоят из прямоугольного корпуса, в котором при помощи пружин устанавливается блок из направляющих пластин. За счет

встроенного клапана расхода, осуществляется плавное регулирование расхода воздуха.

Рисунок 5.2 – Диффузор DLRH 400-6

5.4 Расчет воздуховодов систем вентиляции

Из расчёта воздухообмена выбирается наибольшее значение расхода воздуха, для данной системы L = 1600 м^3/ч. Для промышленных зданий обычно применяются круглые воздуховоды.

Затем считается приблизительный диаметр воздуховода; находится он из формулы:

$$d = \sqrt{\frac{4 \cdot L}{\pi \cdot \nu \cdot 3600}}\text{, м} \qquad (5.3)$$

где V = 12 м/с – скорость воздуха в магистрали;

$$d = \sqrt{\frac{4 \cdot 1600}{3{,}14 \cdot 12 \cdot 3600}} = 0{,}22 \text{ м}/c$$

Из стандартного ряда принимаем воздуховод с диаметром равным d = 250 мм.

5.5 Расчет и подбор дефлекторов

Дефлектор – вытяжное устройство, устанавливаемое на крыше здания для отсоса вентиляционного воздуха и вредностей.

Поток воздуха в нем вычисляется по формуле:

$$Q_{деф} = \frac{V_{в.з}}{n}, м^3/ч; \quad (5.4)$$

где $V_{в.з}$ – двукратный объем верхней зоны помещения, $м^3/ч$

n- количество дефлекторов;

$$Q_{деф} = \frac{324}{1} = 324\ м^3/ч$$

Площадь поперечного сечения дефлектора рассчитывается:

$$S_{деф} = \frac{Q_{деф}}{V_{деф} \cdot 3600}, м^2; \quad (5.5)$$

$$S_{деф} = \frac{\pi d^2}{4}, м^2; \quad (5.6)$$

Приравниваем оба уравнения. Получаем:

$$d = \sqrt{\frac{4Q_{деф}}{V_{деф} \cdot \pi \cdot 3600}}; \quad (5.7)$$

$$d = \sqrt{\frac{4 \cdot 324}{0{,}6 \cdot 3{,}14 \cdot 3600}} = 0{,}44\ м$$

Принимаем к установке дефлектор Д 500.

Таблица 5.3 – Технические характеристики дефлектора Д 500

Диаметр воздуховода, мм	Высота, мм	Размеры цилиндра, мм		Масса, кг
		Диаметр	Высота	
500	951	950	550	31,1

Дефлектор вентиляционный - узел естественной системы вентиляции, предназначен для усиления тяги в воздуховодах за счет использования ветрового напора. Ветер, набегая на дефлектор, создает внутри цилиндрической оболочки зону пониженного давления, способствующую работе вытяжной системы.

Рисунок 5.3 – Дефлектор Д 500

5.6 Расчет и подбор фильтров

В приточных установках первыми по ходу воздуха устанавливаются воздушные фильтры, что позволяет предохранить поверхность последующих технологических блоков от загрязнения пылью, а также не допустить попадание пыли в обслуживаемые помещения.

Фильтры устанавливаются для предотвращения загрязнения поверхности кондиционируемых аппаратов, которые расположены после блоков фильтров, и внутренних поверхностей помещения.

Для получения требуемого класса чистоты, приточный воздух должен проходить многоступенчатую очистку. В зависимости от требуемого класса чистоты, устраивается последовательное прохождение приточного воздуха через фильтры различной эффективности очистки.

Работа воздушных фильтров характеризуется следующими показателями: эффективностью очистки, пылеёмкостью, воздушной удельной нагрузкой.

Для оценки эффективности работы фильтра вычисляют время его работы.

Объект кондиционирования расположен в районе с очень сильно загрязненной атмосферой с концентрацией пыли $C_{вх} = 3$ мг/м^3.

Рекомендуемая концентрация пыли в воздухе после инфильтрации должна быть не больше 0,1 мг/м^3.

$$C_{ВЫХ} = \frac{C_{ВХ} - (A_М \cdot C_{ВХ})}{100}, мг/м^3; \quad (5.8)$$

где $A_м$ – эффект очистки воздуха в фильтрах, %, зависит от класса очистки.

По эффективности действия (фильтрующей способности) воздушные фильтры подразделяются на три класса: грубой очистки, тонкой очистки, высокоэффективные HEPA-фильтры.

В качестве первой ступени фильтрации применяем ячейковые фильтры. Класс очистки G4, $А_м$ = 90 %.

$$C_{ВЫХ} = \frac{3 - (90 \cdot 3\ )}{100} = 0{,}3 мг / м^3;$$

Класс очистки G4 – применим к фильтрам грубой очистки, используемых в помещениях с низкими требованиями очистки воздуха. Предварительно ставятся для очистки в системах вентиляции и центрального кондиционирования. Применяются при эксплуатации компрессоров, холодильных машин в условиях большой запыленности.

В качестве второй ступени фильтрации принимаем фильтрующий блок карманных фильтров.

Принимаем класс очистки F7, $А_м$ = 80 %.

$$C_{ВЫХ} = \frac{0{,}3 - (80 \cdot 3\ )}{100} = 0{,}06 мг / м^3;$$

Класс очистки F7 – применим к фильтрам тонкой очистки воздуха в системах кондиционирования и вентиляции. Применяется в качестве фильтров второй ступени очистки (доочистки).

Время эксплуатации фильтров:

$$\tau_ф = \frac{ПФ \cdot 1000 \cdot F_ф}{(C_{ВХ} - C_{ВЫХ}) \cdot V}, ч; \qquad (5.9)$$

где ПФ – пылеемкость фильтра, г/м^2 ;

$F_ф$ – фронтальная поверхность фильтрующего материала, м^2;

V– пропускная способность фильтра, м^3/ч.

Фильтры подбираем из списка производителя фирмы " Airfilter".

Для ячейковых фильтров:

$$\tau_ф = \frac{5000 \cdot 1000 \cdot 1{,}2}{(3 - 0{,}3\ ) \cdot 1860} = 1195 ч;$$

Для карманных фильтров:

$$\tau_ф = \frac{400 \cdot 1000 \cdot 6{,}8}{(0{,}3 - 0{,}06\ ) \cdot 1860} = 6093 ч;$$

Выбираю четыре воздушных фильтров Airfilter. Airfilter оснащены круглыми уплотнительными соединениями и рычажными замками, предназначен для очистки воздуха в системах приточной вентиляции и кондиционирования.

5.7 Подбор вентилятора для механической вытяжной системы

Вентилятор выбирается по расходу (L) и давлению ($\Delta P_в$). Давление считается по следующей формуле:

$$\Delta P_в = 1{,}2 \cdot P_{мн}, \text{ кгс/м}^2 ; \quad (5.10)$$

где $P_{мн}$ – давление в магистральном направлении, кгс/м^2.

$\Delta P_в = 1{,}2 \cdot 64{,}9 = 77{,}09$ кгс/м^2.

Вытяжка воздуха из помещения осуществляется 2 – мя канальными вентиляторами Systemair K.

Канальные вентиляторы низкого давления предназначены для непосредственной установки в прямоугольный канал вытяжной системы воздуха. Канальный вентилятор используются для перемещения воздуха без твердых, волокнистых и абразивных материалов, а также других невзрывоопасных газовых смесей. Допустимая температура перемещаемого воздуха от -30°С до +40°С.

Рисунок 5.4 – Канальный вентилятор Systemair K

Таблица 5.4 – Технические характеристики канального вентилятора.

Модель	Производительность, ($м^3/ч.$)	Мощность, Вт	ок, А	Частота вращения, мин$^{-1}$	Уровень шума, дБ*	Габаритные размеры	Вес, кг

Продолжение таблицы 5.4

System air K	770	105	457	2553	53.1	160x336x29x26x221x174	4.1

*На расстоянии 3 м

5.8 Вывод по разделу безопасности жизнедеятельности

В результате раздела безопасности жизнедеятельности были подобраны параметры микроклимата, которые поддерживаются системами отопления и вентиляции, была спроектирована система кондиционирования конференц-зала, произведен теплотехнический расчет помещений, аэродинамический расчет воздухораспределения и воздушных сетей по результату которых было подобрано соответствующее оборудование. В ходе выполнения данного раздела была спроектирована планировка конференц зала со схемой расположения вентиляционной сетки, а также расстановки технологического оборудования. Также была представлена схема монтажа наружного и внутреннего блоков кондиционера.

Заключение

В данном дипломном проекте были рассмотрены описание внутренней архитектуры системы, сравнение и описание технологий, использованных при ее создании, проведен анализ средств, а также был отображен конечный интерфейс веб-приложения.

Дипломный проект содержит реализацию клиентской стороны пользовательского интерфейса и программно-аппаратной части сервиса.

Выбор технологий для реализации проекта оказался удачным, весь функционал реализовался без особых трудностей.

Идея проекта заинтересовала многих людей еще на начальной стадии разработки, а значит она актуальна. Разработка веб-портала, рассмотренная в дипломном проекте, – это только первая часть системы AdveCard, которую планирует реализовать компания Web Line.

Список литературы

1 Сайт https://www.digitalocean.com/community/tutorials/linux-apache-mysql-php-lamp-ubuntu-14-04-ru

2 Сайт Описание установки и настройки web-сервера в Ubuntu 14.04 http://senokosov.info/lamp/install-lamp

3 Хортсманн Кей С., Карнелл Гарри. Java 2. Библиотека профессионала. Том 2. Тонкости объектно ориентированного программирования – Вильямс, 2008.1168 с.

4 Котлер Ф., Киллер К.Л. Маркетинг менеджмент 12-е издание – СПб. / Питер, 2007.

5 Молчанов Я. П. Связи организаций с общественностью: экономические, социальные, экологические аспекты: Учеб. Пособие / РХТУ им. Д.И. Менделеева, М., 2002.

6 Левенчук А. Интернет предлагает решения для корпораций / Рынок ценных бумаг 1996.

7 Алтуховьев Д., Ровенова Н. В. Свой сервер в Internet. / Планета Internet. - 1997. - N10.

8 Бабушкин М., Королев В. Как правильно организовать свой Web-сервер. / Мир Internet. - 1997. - N3.

9 Гринфельд М., Кенигфест Г. Реклама и Public Relations в сети Internet. - 1997. - N4.

10Булгарин М. PiR в Интернет. Паблисити, имидж, реклама, паблик релейшенс.-СПб. / АТА"БОЛГАР", 1999.

11 Галкин С. Бизнес в Интернет.- М. / Изд-во "Центр", 1998.

12 Сайт https://docs.joomla.org/J3.x:Installing_Joomla/ru

13 Сайт https://ru.wikipedia.org/wiki/PostgreSQL

14 Сайт https://ru.wikipedia.org/wiki/Joomla!

15 Сайт http://dic.academic.ru/dic.nsf/ruwiki/22191

16 Сайт http://boxapp.net/blog/joomla/sozdanie-ustanovochnogo-paketa-dlya-rasshireniya-joomla-3/

17 Метод. указания к выполнению дипломного и курсового проектов для студентов заочной формы обучения "Расчет воздухообмена в помещениях здания для кондиционирования и вентиляции воздуха". 2006 г.

18 Белова Е. М. Центральные системы вентилирования воздуха в зданиях. — М.: Евроклимат, 2006. — 640 с: ил.

19 Асаньев ВЛ., Балуева Л.Н., Городов А.К., Гальперин А.Д., Еремин М.Ю., Звягтищева СМ, Мурашков В.П., Седова И.В. Системы и кондиционирования и вентиляции. Теория и практика. 2001,416 с Третье издание.

20 СНиП 2.04.05-91*. Отопление, вентиляция и кондиционирование. М.: ГУП ЦПП, 2000.

Приложение А

Текст программы

```
//Пример JAVA сервлета

import static java.lang.Boolean.TRUE;
import java.util.Random;
import java.util.logging.Level;
import java.util.logging.Logger;

import org.json.simple.JSONObject;
import org.json.simple.parser.JSONParser;
import org.json.simple.parser.ParseException;

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

/**
 *
 * @author akirk
 */
public class Users {
    private String type;
    private String countryFormat;
    private String countryValue;
    private String phoneNumberFormat;
    private String phoneNumberValue;
    private String session;

    Users(String request){
        try {
            JSONParser jsonParser = new JSONParser();
            JSONObject jsonObject;
            jsonObject = (JSONObject) jsonParser.parse(request);
            JSONObject requestObj = (JSONObject) jsonObject.get("REQUEST");
            type = (String) requestObj.get("type");
```

```
            session = (String) requestObj.get("session");

            JSONObject countryObj = (JSONObject) requestObj.get("country");
            countryFormat = (String) countryObj.get("format");
            countryValue = (String) countryObj.get("value");

            JSONObject phoneNumberObj = (JSONObject) requestObj.get("phoneNumber");
            phoneNumberFormat = (String) phoneNumberObj.get("format");
            phoneNumberValue = (String) phoneNumberObj.get("value");
        } catch (ParseException ex) {
            Logger.getLogger(Users.class.getName()).log(Level.SEVERE, null, ex);
        }
    }

    public String getCountryFormat() {
        return countryFormat;
    }

    public String getCountryValue() {
        return countryValue;
    }

    public String getType() {
        return type;
    }

    public String getPhoneNumberFormat() {
        return phoneNumberFormat;
    }

    public String getPhoneNumberValue() {
        return phoneNumberValue;
    }

    public String getSession() {
        return session;
    }
```

```
   public JSONObject action(){
   JSONObject jsonObject = new JSONObject();
   JSONObject responseObj = new JSONObject();
      if(type.equals("registrationStep1"))
      {
      Random rnd = new Random(System.currentTimeMillis());
      int smsCode = 1000 + rnd.nextInt(9999);
      String mes = new String();
      mes = String.valueOf(smsCode) + " - код подтверждения для QDeliX.com";
      Smsc sms = new Smsc();
      String[] res = sms.send_sms(phoneNumberValue, mes, "", "");
      responseObj.put("status","OK");
      responseObj.put("error","0");
      jsonObject.put("RESPONSE", responseObj);
      }
      else{
      responseObj.put("status","ERROR");
      responseObj.put("error","UNDEFINED_TYPE");
      jsonObject.put("RESPONSE", responseObj);
      }
   return jsonObject;
   }
   public boolean test(){
      return TRUE;
   }

}//Класс IndividualCards
<?php

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
/**
 * Description of Card
 *
```

```
 * @author akirk
 */
class IndividualCard {
    private $language = '';
    private $isPrivate = 0;
    private $firstName = '';
    private $lastName = '';
    private $patronymic = '';
    private $gender = '';
    private $activityField = '';
    private $speciality = '';
    private $workPlace = '';
    private $position = '';
    private $description = '';
    private $workingHours = '';
    private $telephone = '';
    private $email = '';
    private $facebook = '';
    private $twitter = '';
    private $vkontakte = '';
    private $instagram = '';
    private $linkedin = '';
    private $googleplus = '';
    private $youtube = '';
    private $skype = '';
    private $address = '';
    private $activityArea = '';

    function query() {
    return "INSERT
INTO
  `wnwcl_individualCards`(
    `id`,
    `language`,
    `isPrivate`,
    `firstName`,
    `lastName`,
    `patronymic`,
```

```
    `gender`,
    `activityField`,
    `speciality`,
    `workPlace`,
    `position`,
    `description`,
    `workingHours`,
    `telephone`,
    `email`,
    `address`,
    `facebook`,
    `twitter`,
    `vkontakte`,
    `instagram`,
    `linkedin`,
    `googleplus`,
    `youtube`,
    `skype`,
    `activityArea`
  )
VALUES(
  NULL,
  '" . $this->language ."',
  '" . $this->isPrivate ."',
  '" . $this->firstName ."',
  '" . $this->lastName ."',
  '" . $this->patronymic ."',
  '" . $this->gender ."',
  '" . $this->activityField ."',
  '" . $this->speciality ."',
  '" . $this->workPlace ."',
  '" . $this->position ."',
  '" . $this->description ."',
  '" . $this->workingHours ."',
  '" . $this->telephone ."',
  '" . $this->email ."',
  '" . $this->address ."',
  '" . $this->facebook ."',
```

Продолжение Приложения А

```
  '" . $this->twitter ."',
  '" . $this->vkontakte ."',
  '" . $this->instagram ."',
  '" . $this->linkedin ."',
  '" . $this->googleplus ."',
  '" . $this->youtube ."',
  '" . $this->skype ."',
  '" . $this->activityArea ."'
);";


//return "INSERT INTO 'jos_individualCards' ('id', 'language', 'isPrivate', 'firstName', 'lastName', 'patronymic', 'gender', 'activityField', 'speciality', 'workPlace', 'position', 'description', 'workingHours', 'telephone', 'email', 'address', 'facebook', 'twitter', 'activityArea') VALUES (NULL," + $this->language + "," + $this->isPrivate + "," + $this->firstName + "," + $this->lastName + "," + $this->patronymic + "," + $this->gender + "," + $this->activityField + "," + $this->speciality + "," + $this->workPlace + "," + $this->position + "," + $this->description + "," + $this->workingHours + "," + $this->telephone + "," + $this->email + "," + $this->address + "," + $this->facebook + "," + $this->twitter + "," + $this->activityArea + ");";
    }

    function getLanguage() {
        return $this->language;
    }

    function getIsPrivate() {
        return $this->isPrivate;
    }

    function getFirstName() {
        return $this->firstName;
    }

    function getLastName() {
        return $this->lastName;
    }
```

```
    function getPatronymic() {
        return $this->patronymic;
    }

    function getGender() {
        return $this->gender;
    }

    function getActivityField() {
        return $this->activityField;
    }

    function getSpeciality() {
        return $this->speciality;
    }

    function getWorkPlace() {
        return $this->workPlace;
    }

    function getPosition() {
        return $this->position;
    }

    function getDescription() {
        return $this->description;
    }

    function getWorkingHours() {
        return $this->workingHours;
    }

    function getTelephone() {
        return $this->telephone;
    }

    function getEmail() {
```

```
        return $this->email;
    }

    function getFacebook() {
        return $this->facebook;
    }

    function getTwitter() {
        return $this->twitter;
    }

    function getVkontakte() {
        return $this->vkontakte;
    }

    function getInstagram() {
        return $this->instagram;
    }

    function getLinkedin() {
        return $this->linkedin;
    }

    function getGoogleplus() {
        return $this->googleplus;
    }

    function getYoutube() {
        return $this->youtube;
    }

    function getSkype() {
        return $this->skype;
    }

    function getAddress() {
        return $this->address;
    }
```

```
    function getActivityArea() {
        return $this->activityArea;
    }

    function setLanguage($language) {
        $this->language = $language;
    }

    function setIsPrivate($isPrivate) {
        $this->isPrivate = $isPrivate;
    }

    function setFirstName($firstName) {
        $this->firstName = $firstName;
    }

    function setLastName($lastName) {
        $this->lastName = $lastName;
    }

    function setPatronymic($patronymic) {
        $this->patronymic = $patronymic;
    }

    function setGender($gender) {
        $this->gender = $gender;
    }

    function setActivityField($activityField) {
        $this->activityField = $activityField;
    }

    function setSpeciality($speciality) {
        $this->speciality = $speciality;
    }

    function setWorkPlace($workPlace) {
```

```
        $this->workPlace = $workPlace;
    }

    function setPosition($position) {
        $this->position = $position;
    }

    function setDescription($description) {
        $this->description = $description;
    }

    function setWorkingHours($workingHours) {
        $this->workingHours = $workingHours;
    }

    function setTelephone($telephone) {
        $this->telephone = $telephone;
    }

    function setEmail($email) {
        $this->email = $email;
    }

    function setFacebook($facebook) {
        $this->facebook = $facebook;
    }

    function setTwitter($twitter) {
        $this->twitter = $twitter;
    }

    function setVkontakte($vkontakte) {
        $this->vkontakte = $vkontakte;
    }

    function setInstagram($instagram) {
        $this->instagram = $instagram;
    }
```

```
    function setLinkedin($linkedin) {
        $this->linkedin = $linkedin;
    }

    function setGoogleplus($googleplus) {
        $this->googleplus = $googleplus;
    }

    function setYoutube($youtube) {
        $this->youtube = $youtube;
    }

    function setSkype($skype) {
        $this->skype = $skype;
    }

    function setAddress($address) {
        $this->address = $address;
    }

    function setActivityArea($activityArea) {
        $this->activityArea = $activityArea;
    }

}

?>
```