УНИВЕРСИТЕТ «УРБАН»

**ДИПЛОМНАЯ РАБОТА**

**на тему: Разработка веб-приложения для работы с абонентами и приборами учета воды на действующем предприятии ООО «Горводоканал»**

Автор Цитцер Евгений Викторович

г.Черепаново, 2024 г.

**Оглавление:**

1. Введение………………………………………………………………………..3
   1. Обоснование выбора темы……………………………………………..3
   2. Определение цели и задачи работы……………………………………3
2. Основные понятия и определения……………………………………………4
3. Методы и подходы к разработке……………………………………………..4
   1. Выбор подходящего фреймворка для разработки……………………4
   2. Архитектура веб-приложения………………………………………….4
   3. Обеспечение безопасности……………………………………………..4
4. Обзор популярных инструментов для разработки веб-приложений

на Python………………………………………………………………………..5

* 1. Django……………………………………………………………………5 Fastapi……………………………………………………………………5
  2. Flask……………………………………………………………………...5

1. Проектирование приложения…………………………………………………6
   1. Планирование и анализ требований…………………………………...6
   2. Основные требования…………………………………………………..6
   3. Технические требования……………………………………………….7
2. Разработка приложения в соответствии с созданной документацией……..7
3. Заключение………………..…………………………………………………..11
   1. Анализ результатов работы……………………………………………11
   2. Планы к практическому применению разработки…………………...11
4. Введение
   1. Обоснование и выбор темы.

В настоящее время, когда без преувеличения, все сферы экономики подвержены цифровизации, государственные услуги уже в подавляющем большинстве можно получить парой кликов мышью, существуют небольшие предприятия в малых городах, которые всё еще осуществляют связь с клиентами посредством телефонной связи и электронной почты. А основная масса услуг предоставляется только при личном посещении кабинетов. На предприятии, где я в настоящее время работаю, учет абонентов и приборов учета до сих пор ведется в таблицах Excel. Абонент зачастую узнает об окончании срока поверки прибора учета только после начисления завышенных сумм за водоснабжение по нормативам. По закону, ресурсоснабжающие предприятия обязаны иметь свой сайт. У нашего предприятия имеется сайт, который создан на базе конструктора и не имеет никакого функционала, кроме информационного. Поэтому мой выбор пал на тему разработки веб-приложения, которое будет включать в себя информационную составляющую о деятельности предприятия, а также функциональную составляющую, включающую в себя возможность создания базы абонентов, их приборов учета, а также отслеживания сроков поверки приборов учета и информирования.

Таким образом, выбор темы обусловлен желанием разработать реально действующее веб-приложение, которое сможет повысить эффективность работы предприятия, систематизирует работу с приборами учета воды, улучшит коммуникативную составляющую между предприятием и абонентами.

* 1. Определение цели и задачи работы.

Целью работы является разработка реально действующего веб-приложения по работе с абонентами и их приборами учета воды. Для достижения поставленной цели нужно реализовать следующие задачи:

* Разработать видимую часть сайта (frontend), где будет доступна общая информация о предприятии, контакты, действующие тарифы на услуги, качество предоставляемых услуг.
* Разработать функциональную часть приложения (backend), в которой будет возможность наполнять базу данных абонентов и приборов учета, вести контроль за сроками поверки приборов учета и информировать абонентов об окончании этих сроков посредством SMS-рассылки.
* Создать личный кабинет абонента, где он самостоятельно может отслеживать состояние приборов учета.
* Обеспечить возможность дальнейшего расширения функционала веб-приложения.

1. Основные понятия и определения

* Frontend - это разработка функциональности и пользовательского интерфейса, работающих на клиентской стороне приложения или веб-сайта.
* Backend - это программно-аппаратная часть сервиса. Это набор средств, с помощью которых происходит реализация логики веб-сайта.
* Фреймворк (Framework): - это программная платформа, которая предоставляет готовые компоненты и инструменты для разработки приложений. В контексте веб-приложений часто используются Django, FastAPI или Flask.
* Веб-приложение (Web Application): - это программное приложение, которое работает на веб-сервере и доступно через браузер.
* Прибор учета воды – это прибор, предназначенный для измерения и учёта объёма воды, проходящего по водопроводу в месте его установки.

1. Методы и подходы к разработке
   1. Выбор подходящего фреймворка для разработки.

Для реализации поставленной цели в рамках данной работы рациональнее всего произвести разработку на фреймворке Django. Так как приложение будет работать с большими базами данных, модель MVT в Django в полной мере подходит для этих задач. Также полностью отвечают целям работы такие инструменты Django как формы регистрации, системы администрирования, встроенные шаблоны.

* 1. Архитектура веб-приложения.

Фронтенд будет разработан с использованием HTML, CSS и JavaScript. Фронтенд будут составлять Главная страница, О предприятии, Контакты, Тарифы, Качество воды. Также страница авторизации абонента, страница личного кабинета абонента.

Бэкенд будет реализован на Python с использованием фреймворка Django. Будет представлен логикой авторизации, определением моделей баз данных и логикой их заполнения, логика работы с приборами учета, определение маршрутов и др.

* 1. Обеспечение безопасности.

Для исключения возможности регистрации постороннего абонента по определенному адресу, регистрация и создание личного кабинета будет возможно только администратором веб-приложения при личном присутствии абонента. Логин и первичный пароль будут переданы абоненту. Для посещения личного кабинета предусмотрена система авторизации по установленным логину и паролю.

1. Обзор популярных инструментов для разработки веб-приложений

на Python.

4.1. Django.

Django —фреймворк для веб-разработки на Python.

Основные возможности:

· Встроенная страница администрирования для управления данными.

· ORM (Object-Relational Mapping) для работы с базами данных.

· Поддержка аутентификации пользователей и управления доступом.

· Мощная система маршрутизации URL.

· Шаблонизатор для создания HTML-шаблонов.

Особенности:

Идеально подходит для больших и сложных проектов благодаря своей полной экосистеме.

4.2. Fastapi.

Это современный, высокопроизводительный фреймворк для создания API с Python 3.7+ на основе стандартов OpenAPI и JSON Schema.

Основные возможности:

· Высокая производительность благодаря использованию асинхронного программирования (async/await).

· Автоматическая генерация интерактивной документации API.

· Поддержка валидирования и аннотаций типов данных.

· Интеграция с различными библиотеками и инструментами, такими как Pydantic и SQLAlchemy.

Особенности:

Часто используется для создания API и микросервисов.

4.3. Flask.

Это фреймворк для веб-разработки на Python. Обеспечивает простоту и гибкость.

Основные возможности:

Минималистичный подход: только основные компоненты без лишних зависимостей.

Высокая гибкость для добавления нужных библиотек и модулей.

Jinja2 для шаблонов и Werkzeug для маршрутизации и обработки запросов.

Особенности:

Подходит для небольших и средних проектов, где требуется большая гибкость.

1. Проектирование приложения.
   1. Планирование и анализ требований

Выбор фреймворка и инструментов: Определение наиболее подходящего фреймворка (например, Django, Flask или FastAPI) и инструментов для разработки.

Определение структуры приложения: Разработка схемы архитектуры, включающей фронтенд, бэкенд, базу данных.

Разработка прототипа

Реализация основного функционала: Создание базовой версии приложения с минимально необходимым функционалом для наполнения базы данных и демонстрации страниц сайта.

Разработка ключевого функционала

Создание личного кабинета абонента и системы авторизации. Реализация функционала по контролю за сроками поверки приборов учета и SMS-рассылке.

* 1. Основные требования.

• Создать модели баз данных адресов и абонентов и приборов учета (абоненты и приборы учета могут быть в одной модели).

• Обеспечить необходимую связь между моделями.

• Регистрация нового абонента должна производиться из панели администратора.

• При регистрации абонента предусмотреть автоматическое выпадающее меню с базой сохраненных адресов. Если адрес не существует, то при сохранении его в базе абонентов, он должен автоматически сохраняться в базе адресов.

• При регистрации абонентов сохранять действующий номер сотового телефона абонента.

• При регистрации прибора учета воды в базе абонентов сохранять паспортную дату поверки и следующую дату поверки. При приближении следующей даты поверки за 60 дней отправлять абоненту SMS-сообщение о необходимости проведения поверки прибора учета.

• Реализовать периодическую проверку на предмет наступления необходимого момента для выполнения задачи по отправке SMS-сообщения абоненту с помощью библиотеки Celery.

• Создать страницу личного кабинета абонента. Вход в личный кабинет обеспечить посредством ввода логина и пароля, которые были созданы администратором при регистрации абонента. С учетом того, что информация в личном кабинете не является конфиденциальной, не содержит личных данных абонента и редактируется администратором, функционал по смене пароля можно не предусматривать. В дальнейшем это можно изменить.

• В личном кабинете абонент может увидеть информацию, сохраненную в базе абонентов по id, присвоенному при регистрации.

• На именованных страницах сайта расположить любую соответствующую информацию, не являющуюся конфиденциальной.

* 1. Технические требования.

• Фронтенд: использовать HTML, CSS, JavaScript, возможно Bootstrap.

• Бэкенд: Python с использованием фреймворков Django.

• База данных: использование базы данных (в данном проекте PostgreSQL) для хранения данных определенных созданными моделями.

1. Разработка приложения в соответствии с созданной документацией.

Планирование разработки:

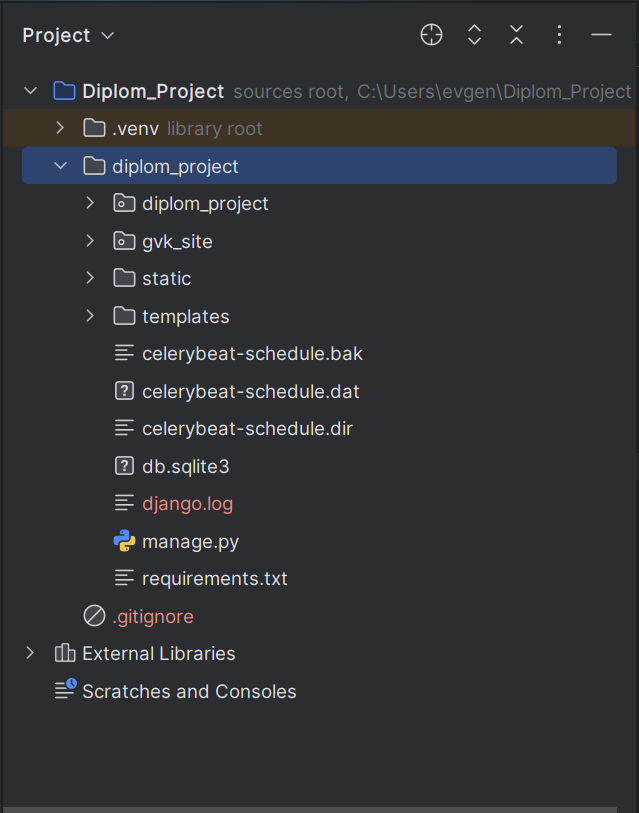
Разработка была разделена на несколько основных этапов: проектирование интерфейса, реализация серверной логики, тестовое заполнение баз данных. Отслеживание результата производилось с помощью встроенного в Django локального сервера.

Разработка:

* 1. Создание проекта.

Создан Django-проект с названием diplom\_project. В данном проекте создано приложение gvk\_site.

Структура проекта:

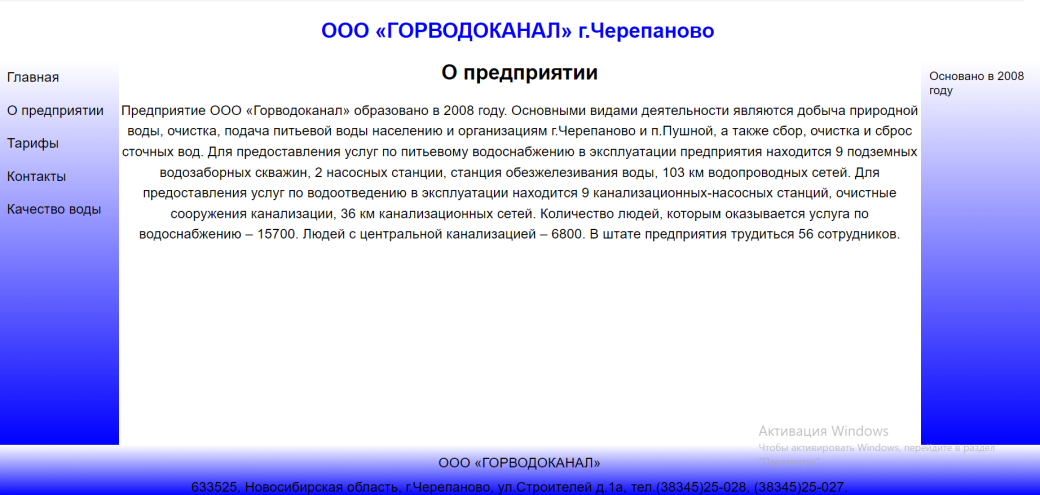


* 1. Фронтенд. Шаблоны разработаны на основе html и CSS. Всего создано 7 шаблонов страниц приложения.

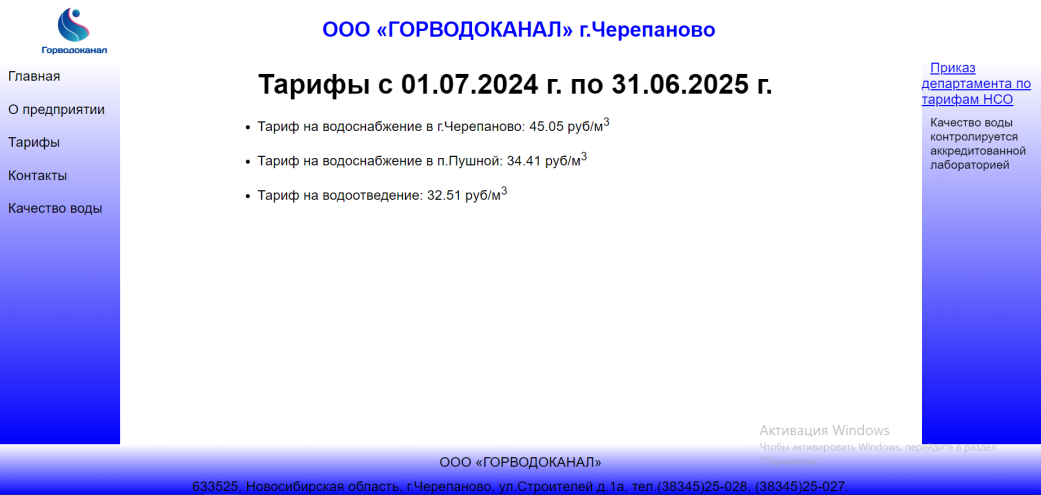
Главная страница:



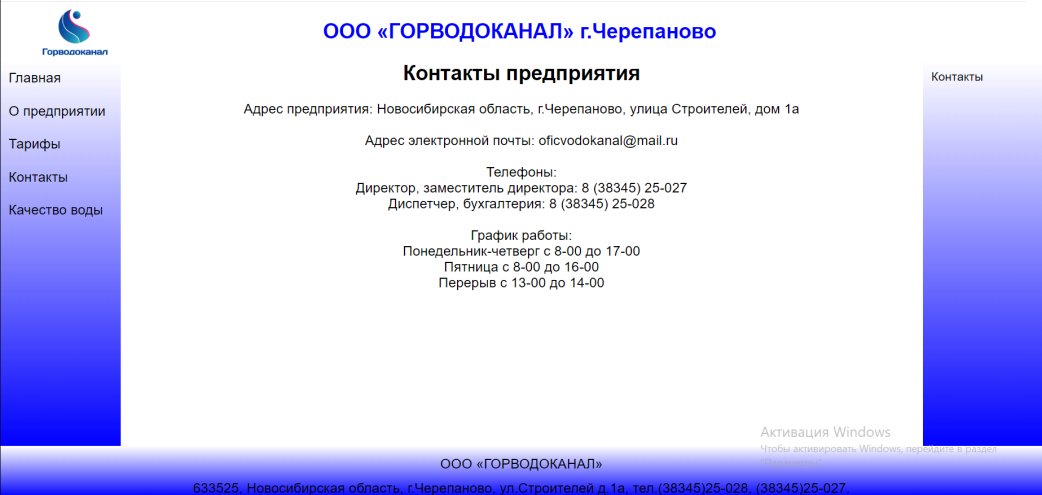
Страница «О предприятии»:



Страница «Тарифы»:



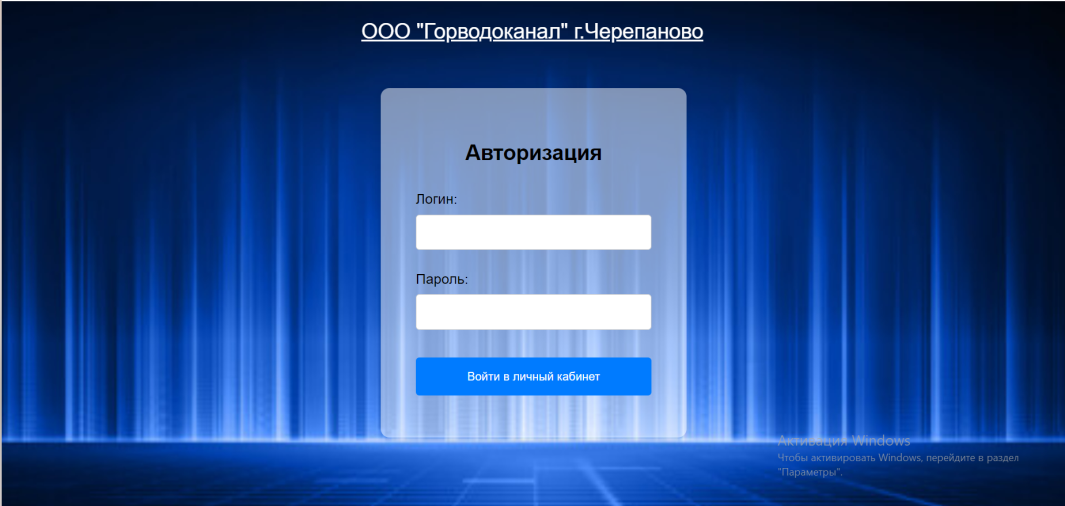
Страница «Контакты»:



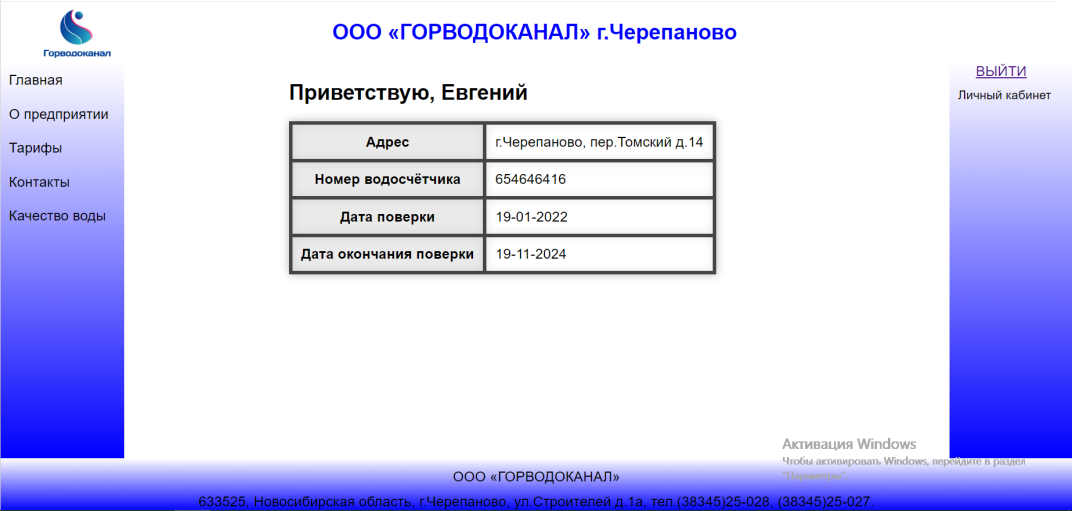
Страница «Качество воды»:



Страница авторизации:



Страница «Личный кабинет»

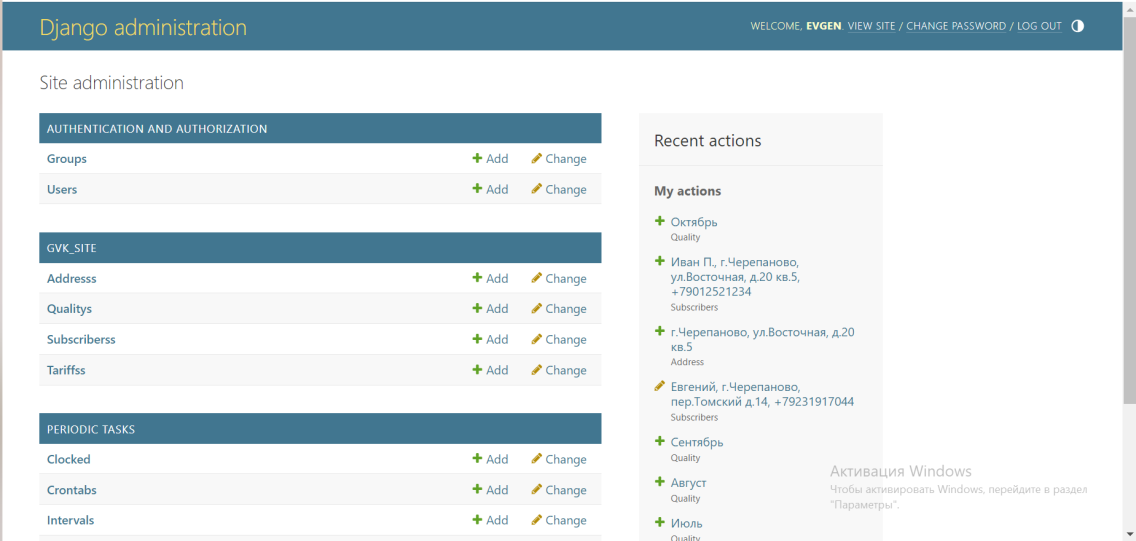


* 1. База данных.

В качестве базы данных было создано 4 модели: Абоненты (Subscribers), Адрес (Address), Тарифы (Tariffs), Качество воды (Quality). Модель Subscribers имеет внешний ключ (Foreign key) к полю address модели Address. При регистрации абонента в базе данных Subscribers можно выбрать существующий в модели Address адрес абонента. В случае, если вводимого адреса не существует, то он при регистрации добавляется в Address.

* 1. Администрирование.

Данные в созданные базы данных имеет право вносить и корректировать только администратор приложения. Для этого был создан superuser. На странице администратора зарегистрированы модели баз данных. Для этого в модуле admin были созданы соответствующие функции.



* 1. Авторизация.

После регистрации абонента администратором, в модель Subscribers сохраняются информация о логине и пароле абонента. Для авторизации абонента на основе Django-form создана форма авторизации LoginForm. При вводе логина и пароля абонент попадает на страницу личного кабинета, в которой содержится информация о его адресе, номере прибора учета воды, дате поверки и окончания поверки этого прибора.

* 1. Расписание и SMS-уведомления.

Основной функционал приложения на момент разработки заключается в извещении абонентов об истечении срока поверки прибора учета воды. Для реализации этого функционала применены следующие решения: установлена библиотека Celery - это система управления задачами, она работает совместно с брокером сообщений для выполнения асинхронных задач. В данном проекте задействован брокер RabbitMQ. Для отправки SMS-сообщений был создан и настроен аккаунт на сервисе SMSAero. Для Celery была создана задача в модуле task. В функции задачи входит 1 раз в сутки в 8-00 часов по московскому времени проверять базу данных Subscribers и сравнивать дату окончания поверки прибора учета (verify\_period) с текущей датой. Если от текущей даты до verify\_period осталось 60 дней, то функция отправляет SMS-сообщение с напоминанием о необходимости поверки прибора на номер телефона, который указан в поле telephone\_number абоненту с тем же id.

* 1. Тестирование и логирование.

В модуле test приложения были разработаны функции тестирования представлений на основе класса TestCase библиотеки Unittest. Для имитации работы браузера во время тестирования использовался класс Client из модуля django.test.client. Тестирование информационных страниц сайта проводилось на получение нужного ответа на HTTP-запросы (ожидался код 200), тестирование на получение нужного шаблона, а также тестирование передачи контекста данных. Для реализации тестирования передачи контекста данных в методе класса setUp были созданы тестовые объекты базы данных. В тесте страницы авторизации проверяли успешную аутентификацию с правильными учетными данными. Успешный вход должен перенаправить пользователя на страницу личного кабинета (ожидался код 302) и сохранить идентификатор подписчика в сессии. Кроме этого проверяли попытки входа с неправильными учетными данными и несуществующим пользователем.

Для подключения логирования были применены соответствующие настройки в модуле settings. Применено логирование в файл с уровнями Info и выше. Далее установил импортировал библиотеку logging и создал объект logger, который присвоил логируемым функциям для записи логов.

1. Заключение.
   1. Анализ результатов работы.

В результате работы над для дипломным проектом было разработано веб-приложение, полностью соответствующее цели и задачам проекта. При запуске веб-приложения на локальном сервере имеется возможность переходить по ссылкам на соответствующие страницы сайта, размещать на них любую информацию, а также информацию из созданных баз данных. Осуществляются функции по регистрации абонентов персоналом, обладающим правами администратора, авторизации абонентов с переходом в личный кабинет. Реализована функция SMS-оповещения абонентов об истечении срока поверки прибора учета воды. Тестовое сообщение было отправлено из кода приложения и получено на личный телефон. Применены методы контроля за правильной работой приложения: тестирование и логирование.

* 1. Планы к практическому применению разработки.

Выбор темы дипломной работы был обусловлен желанием применить полученные в результате обучения знания для реализации практически полезного приложения. Следует понимать, что в основном разработка не уникальна, и существует множество приложений для работы с базами данных абонентов коммунальных услуг. Однако в реальной работе я часто сталкиваюсь с тем, что абоненты, несмотря на их обязанность следить за своими приборами учета воды, не знают, что у них закончился срок поверки этих приборов, и обращаются в нашу службу, когда уже получают счета с расчетом оплаты по нормативам. Так как в распоряжении нашей службы имеется база данных об этих абонентах, то и имеется возможность импортировать её в разработанное приложение и осуществлять предупреждение абонентов посредством SMS-рассылки. Кроме этого, в планах создать функционал по фиксированию показаний прибора учета в личном кабинете абонента. Это позволит вычислить потребленный объем воды, и далее, используя данные из базы тарифов, формировать счёт для оплаты за потребленные услуги. Возможность оплаты услуг через данное веб-приложение не рассматривается, так как нужно обеспечивать более высокие уровни защиты.