# 2 РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ

## 2.1 Описание технологий разработки программного средства

Разработка веб-приложения для визуализации мониторинга качества воды потребовала тщательного выбора современных технологий, обеспечивающих высокую производительность, удобство взаимодействия с пользователем и надежность хранения данных. Система построена по трехзвенной архитектуре, включающей фронтенд, бэкенд и базу данных, каждая часть которой реализована с использованием оптимальных технологических решений.

Для разработки пользовательского интерфейса был выбран React.js – современный JavaScript-фреймворк. React.js – это богатая экосистема дополнительных библиотек и инструментов, а также уже не первый год существование на рынке, что дает свои плюсы – большое количество материалов на просторах интернета. Фронтенд часть системы также использует Leaflet.js для интерактивного взаимодействия с картами. С помощью такой библиотеки можно выбрать в качестве основного слоя любую из доступных карт (будь то Google Maps, Yandex Maps или Open Street Maps), на который можно добавляет различные объекты. Например, такими объектами могут быть пины на карте или, как в нашем случае, гексагоны с цветовой разметкой. Это фреймворк, который предоставляет набор функций разработана на языке программирования Java с использованием фреймворка Spring Boot. Также используется библиотека D3 для создания гексагонов на карте.

Серверная часть приложения разработана на Python с фреймворком Django для взаимодействия по REST API. Также среди используемых библиотек есть такие, как Selenium, для возможности обрабатывать данные с сайтов минского водоканала, а также psycopg2 для взаимодействия с PostgreSQL. Серверная часть также отвечает за подсчет средних значений по гексагону. Эта логика специально отселена в отдельную компоненту – крон таску, которая рассчитывает значения для всех гексагонов и которая реализована с помощью apsheduler. ApSheduler – это технология, которая позволяет периодически запускать действие, указанное в специальных функциях. Так, например, в нашем случае мы запускаем периодически тяжелые расчеты (сбор информации о всех улицах города Минска и параметрах воды в домах), после чего сохраняем всю полезную информацию внутрь базы данных. Тут стоит отметить, что из-за асинхронной схемы взаимодействия между частью с расчетами и потенциальными клиентами сервиса могут возникать повторные расчеты – если выполнение задачи по крону еще не завершено, а пользователь уже хочет получить данные.

Для хранения данных была выбрана реляционная база данных PostgreSQL. Этот выбор обусловлен ее надежностью, поддержкой сложных запросов и масштабируемостью. PostgreSQL позволяет эффективно хранить и обрабатывать большие объемы данных.

В процессе разработки использовались следующие инструменты: PyCharm, WebStorm и Visual Studio Code в качестве основных сред разработки для написания кода на Python и JS, Git для контроля версий и эффективного управления кодом.

Выбор технологий для разработки системы был основан на их надежности, доступности и скорости разработки. Python и JS обеспечили доступность, легкость в освоении и скорость разработки backend-части, а PostgreSQL — надежное хранение данных. Таким образом, выбранные технологии и инструменты обеспечили эффективную разработку системы, которая соответствует всем поставленным требованиям и готова к дальнейшему масштабированию.

## 2.2 Структура системы

Система веб-приложения для визуализации качества воды строится вокруг центрального модуля обработки и представления данных. Ядром системы выступает серверная часть, реализованная на Python с использованием фреймворка *Django*, который обеспечивает гибкость и простоту разработки API. Сервер взаимодействует с базой данных PostgreSQL, где хранятся как текущие показатели качества воды, так и исторические данные для анализа тенденций. Для обеспечения бесперебойной работы системы реализован механизм автоматического обновления данных через регулярные запросы к API Минскводоканала.

Система состоит из нескольких модулей, которые взаимодействуют между собой через REST API. Это обеспечивает масштабируемость и модульность системы, позволяя легко добавлять новые функции или изменять существующие без необходимости переписывания всей системы. Основные модули включают:

1 Веб-интерфейс, предназначенный для пользователей, позволяющий взаимодействовать с картой и вводить адреса интересующих их мест. Пользователь может указать как улицу, так и номер интересующего гексагона. Также пользователь может сравнивать две даты по качеству воды.

2 Модуль обработки данных и формирования агрегируемых параметров по гексагону (серверная часть) отвечающий за обработку данных, полученных из минского водоканала. Сервер рассчитывает средние показатели качества воды в гексагонах и сохраняет такую информацию в PostgreSQL. Сервер делает это в асинхронном режиме в качестве крон задачи, что позволяет реализовать более эффективный расчет данных на карте.

3 Модуль хранения данных (база данных) обеспечивает хранение всех данных, включая информацию о гексагонах, параметрах воды и результатах анализа. База данных построена на PostgreSQL, что обеспечивает надежность и масштабируемость.

Взаимодействие между модулями организовано следующим образом: сервер в асинхронном режиме рассчитывает полезные для всей системы данные – это параметры воды по гексагонам. Для выполнения этой цели серверу необходимо получить список всех улиц и всех домов города Минска, после чего рассчитать средние показатели качества воды в конкретном гексагоне. Веб-интерфейс отправляет данные в сервер для извлечения информации из PostgreSQL, которая предварительно была посчитана с помощью асинхронной задачи. При отсутствии полезной информации в PostgreSQL (например, если задача была временно отключена или выполняется слишком долго) происходит синхронной расчет информации, запрошенной пользователем.

Пользователи взаимодействуют с системой через интерактивную карту, где могут кликать на цветные гексагоны для просмотра детальных показателей качества воды во всплывающих окнах. В интерфейсе есть настройки отображения данных по временным периодам. Пользователи могут масштабировать карту для детального изучения конкретных районов или просматривать общую картину по всему городу. Дополнительная панель инструментов позволяет переключаться между различными режимами визуализации, включая тепловую карту и сравнительный анализ по месяцам. Все данные обновляются автоматически при изменении информации в источнике, обеспечивая актуальность представленной информации.

Таким образом, система обеспечивает комплексный подход визуализации качества воды. Взаимодействие между модулями через REST API обеспечивает гибкость и масштабируемость системы, что позволяет легко адаптировать ее под нужды конкретных пользователей или организаций.

## 2.3 Алгоритм работы программы

Алгоритм работы программы включает несколько ключевых этапов, начиная с аутентификации пользователя и заканчивая формированием отчетов. Одним из важнейших аспектов системы является алгоритм аутентификации, который обеспечивает безопасный доступ к данным и защиту от несанкционированного доступа.

В приложении применяется следующий алгоритм авторизации пользователя в системе:

1 Пользователь (водитель, диспетчер или администратор) отправляет запрос на аутентификацию, предоставляя свои учетные данные (логин и пароль).

2 Сервер проверяет учетные данные в базе данных. Если данные верны, сервер генерирует два токена: access token и refresh token.

3 Access token имеет ограниченный срок действия (15 минут) и используется для доступа к защищенным ресурсам системы.

4 Refresh token имеет более длительный срок действия (7 дней) и используется для обновления access token после истечения его срока.

5 Оба токена сохраняются в базе данных Redis для быстрого доступа и кэширования.

Для наилучшего описания алгоритма действий всех участников системы была выбрана диаграмма взаимодействия, которая представлена на рисунке 2.1. Данные диаграммы используются, чтобы продемонстрировать взаимодействие актёров во времени и выделить этапы работы системы.

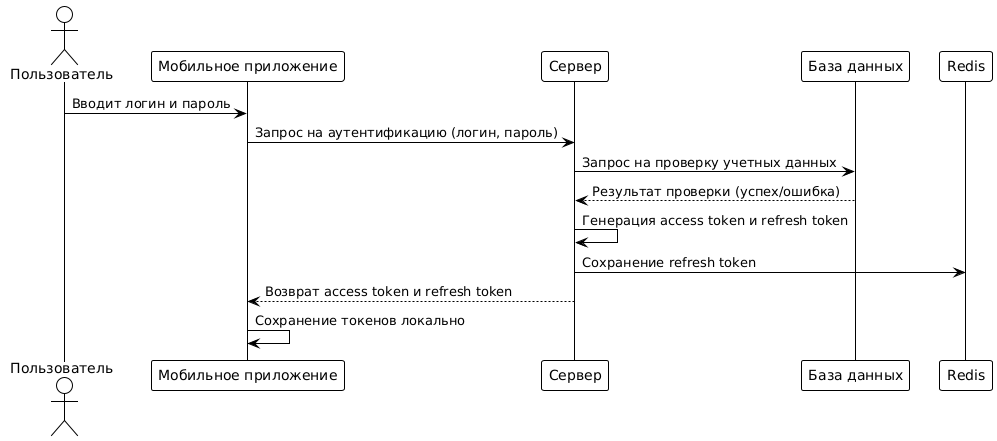


Рисунок 2.1 – Диаграмма взаимодействия при авторизации в системе

Алгоритм получения доступа к защищенным ресурсам системы:

1 При каждом запросе к защищенным ресурсам (например, ввод данных о поездке или запрос отчета) клиентское приложение отправляет access token в заголовке запроса.

2 Сервер проверяет валидность access token, используя секретный ключ для расшифровки. Если токен валиден, запрос обрабатывается.

3 Если access token истек, клиентское приложение отправляет refresh token на сервер для получения нового access token.

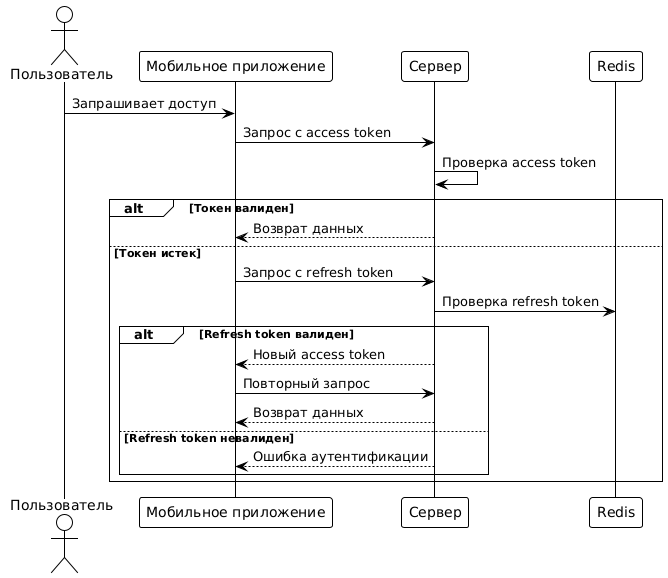


Рисунок 2.2 – Диаграмма взаимодействия при попытке доступа к системе

Алгоритм обновления токенов:

1 Если access token истек, клиентское приложение отправляет refresh token на сервер для получения нового access token.

2 Сервер проверяет refresh token в базе данных Redis. Если токен валиден, генерируется новый access token и отправляется клиенту.

3 Если refresh token также истек, пользователь должен пройти процедуру аутентификации заново.

Алгоритм ввода данных водителем:

1 Водитель вводит данные о поездке через мобильное приложение: название или код рейса, дату начала и окончания поездки, места начала и окончания маршрута, а также время, затраченное на сон.

2 Данные отправляются на сервер через REST API. Сервер проверяет access token в заголовке запроса для подтверждения прав доступа.

3 Если токен валиден, сервер сохраняет данные в базе данных PostgreSQL и начинает обработку.

Алгоритм обработки данных и расчет эффективности водителя:

1 Сервер рассчитывает общую скорость поездки на основе расстояния между точками начала и окончания маршрута и времени, затраченного на поездку.

2 Время сна проверяется на соответствие нормам, установленным законодательством или внутренними правилами компании.

3 На основе этих данных сервер вычисляет итоговую эффективность водителя.

Алгоритм формирования отчета:

1 По запросу от клиента (водителя или диспетчера) сервер извлекает данные из базы данных и формирует отчет.

2 Отчет включает информацию о поездках, скорости, времени сна и эффективности водителя.

3 Отчет отправляется обратно в мобильное приложение или веб-интерфейс для отображения.

Таким образом, алгоритм работы программы обеспечивает безопасный и эффективный процесс аутентификации, обработки данных и формирования отчетов. Использование access и refresh токенов, а также кэширование в Redis позволяют обеспечить высокую производительность и безопасность системы.

## 2.4 Структура базы данных программного средства

База данных веб-приложения для усовершенствования мониторинга качества воды состоит из таблицы hexagons, в которой хранится информация о гексагонах.

Таблица одна и имеет следующую схему: поле hex\_id, которое является главным для поиска, created\_at – timestamp, который отражает момент попадания гексагона в систему для расчета его параметров воды. Такое поле необходимо, чтобы фильтровать данные, которые оказались либо слишком старыми, для текущего промежутка времени, либо слишком новыми, если сейчас пользователь находится в режиме сравнения. Также в таблице присутствует еще два поле: hex\_color и water\_parameters. Где hex\_color – это строка, которая отвечает за то, какой цвет сейчас должен быть у гексагона в шестнадцатеричном формате. Поле water\_parameters содержит все возможные параметры воды для текущего гексагона. Например, запах, жесткость, минерализация.

Для дальнейших доработок, например, при введении уведомлений пользователем через их личный аккаунт, могут быть введение еще несколько соответствующих таблиц. Одной из таких таблиц может быть users, в которой будут хранится email’ы, по которым необходимо отправлять уведомления о плохом качестве воды в их районе. Однако такая схема может быть сложно реализуемой с точки зрения безопасности, так как в базе данных начинают хранится персональные данные пользователей.

Таким образом, база данных системы обеспечивает гибкость и масштабируемость, позволяя эффективно хранить и обрабатывать данные о гексагонах, времени их создания, а также параметрах воды. Такая структура позволяет системе анализировать средние показатели и выбирать временные промежутки для отображения.

## 2.5 Выводы и оценка результатов разработки

Разработка веб-приложения для визуализации мониторинга качества воды успешно завершена. Система полностью соответствует поставленным требованиям и обеспечивает эффективное решение задач сбора, анализа и наглядного представления данных о состоянии водных ресурсов.

Ключевым преимуществом приложения является его специализированный функционал, ориентированный на визуализацию качества воды через гексагональную сетку. Система автоматически загружает и анализирует данные с сайта Минскводоканала, преобразуя табличные показатели в интуитивно понятную цветовую карту. Пользователи могут изучать детализированную информацию по районам, применять фильтры для анализа конкретных параметров воды и сравнивать данные за разные периоды.

В процессе разработки были успешно решены задачи интеграции с внешними источниками данных, проектирования алгоритмов классификации показателей воды и реализации интерактивной карты на основе Leaflet.js и D3.js. Система прошла тестирование на реальных данных, подтвердив точность визуализации и стабильность работы.

Таким образом, разработанное веб-приложение представляет собой эффективный инструмент для мониторинга качества воды, сочетающий удобство использования с мощной аналитической составляющей.

Веб-приложение способствует повышению осведомленности населения и упрощает работу специалистов, ответственных за контроль водных ресурсов. Система готова к внедрению и в будущем может быть масштабирована для охвата других городов или дополнительных параметров анализа, если будет соответствующий источник данных.