Техническое задание на разработку SMS-шлюза

Цель проекта

Разработать систему SMS-шлюза на языке Go, интегрированную с сервисом sms.ru. Система должна включать REST API, очереди для обработки сообщений, а также пользовательский интерфейс для отправки SMS. Проект нацелен на проверку навыков работы с Go, интеграции с внешними API, работы с очередями и разработки пользовательского интерфейса.

Основные требования

1. **Язык разработки:** Go.
2. **API для отправки SMS:** Интеграция с сервисом sms.ru.
3. **Обработка очередей:** Использовать очередь (например, Redis или RabbitMQ) для асинхронной обработки SMS запросов.
4. **REST API и пользовательский интерфейс:** Реализовать REST API и UI для взаимодействия с системой.

Функциональные требования

1. **Интеграция с sms.ru:**
   - Подключение к сервису sms.ru через API ключ.
   - Реализовать отправку SMS через API sms.ru с обработкой всех необходимых параметров (номер телефона, текст сообщения).
   - Обработать все возможные ошибки и статусные ответы от sms.ru.
2. **Пользовательский интерфейс (UI) для отправки SMS:**
   - Разработать простой веб-интерфейс, позволяющий пользователям отправлять SMS.
   - Поля для ввода: номер телефона, текст сообщения.
   - Кнопка отправки, отображение статуса отправки (успех, ошибка).
   - Логирование действий пользователя (успешные и неудачные попытки отправки).
3. **Регистрация и отправка SMS-запросов:**
   - Система должна принимать запросы через UI и REST API.
   - Все входящие сообщения должны сохраняться в базу данных и попадать в очередь для отправки через sms.ru.
4. **Обработка очереди:**
   - Воркер должен забирать сообщения из очереди и отправлять их через sms.ru.
   - В случае ошибки отправки сообщение должно быть возвращено в очередь для повторной обработки с ограничением на количество попыток.
5. **Статус SMS:**
   - Реализовать функционал проверки статуса отправленного сообщения как через UI, так и через API.
6. **Логирование и мониторинг:**
   - Вести лог всех операций, включая успешные и неудачные отправки сообщений.
   - Простой мониторинг состояния системы: количество сообщений в очереди, состояние воркера.

Нефункциональные требования

1. **Безопасность:**
   - Аутентификация для REST API и UI (например, через API ключи или базовую авторизацию).
2. **Производительность:**
   - Обеспечить минимальную задержку при обработке сообщений и работу с высокой нагрузкой.

3. **Качество кода:**
    - Читаемый, поддерживаемый код с комментариями и тестами.
    - Использование Git для контроля версий, итоговый код представить в GitHub/GitLab репозитории.

Требования к окружению

1. **Очередь:** Redis или RabbitMQ.
2. **База данных:** Любая реляционная база данных (например, PostgreSQL).
3. **API sms.ru:** Зарегистрироваться на sms.ru, получить API ключ и настроить интеграцию.
4. **Фронтенд:** Простой интерфейс может быть реализован с использованием HTML, CSS и JavaScript (возможны фреймворки вроде Vue.js или React по желанию).

Критерии оценки

1. Полная и корректная интеграция с sms.ru и корректная обработка API запросов.
2. Работоспособность и правильная обработка очередей.
3. Качество интерфейса, его удобство и функциональность.
4. Соответствие стандартам разработки на Go, наличие тестов и документации.

Заключение

После завершения работ нужно будет выложить исходники проекта в **«github»** в режиме **«public»** и отправить ссылку репозитория на телеграм @khalievl.