МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**
Кафедра математической кибернетики и компьютерных наук

РАЗРАБОТКА СИСТЕМЫ ОПОВЕЩЕНИЯ О СБОЯХ В КОДЕ В РЕАЛЬНОМ ВРЕМЕНИ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 451 группы
направления 09.03.04 — Программная инженерия
факультета КНиИТ
Кузякина Никиты Александровича

Научный руководитель
доцент, к. ф.-м. н. _____ Ю. Н. Кондратова

Зав.кафедрой,
к. ф.-м. н., доцент _____ С. В. Миронов

Саратов 2024

ВВЕДЕНИЕ

В современном мире, где информационные технологии занимают центральное место в различных сферах деятельности, надежность и стабильность программного обеспечения становятся ключевыми аспектами успешного функционирования бизнес-процессов и обеспечения безопасности данных. С постоянным ростом сложности программных продуктов и увеличением объемов кода сбои в его функционировании становятся неизбежным явлением. Это приводит к необходимости оперативного выявления и устранения ошибок с целью минимизации негативного влияния на работу системы. В этом контексте разработка систем оповещения о сбоях в коде в реальном времени приобретает особую важность.

Целью данной дипломной работы является разработка системы оповещения о сбоях в коде, способной интегрироваться в код на любом языке программирования и своевременно оповещать разработчиков о сбоях.

Для достижения данной цели были поставлены следующие задачи:

- изучение работы подобных систем;
- разработка базы данных для хранения информации о системах;
- разработка подсистемы авторизации в системе с помощью Spring Security;
- разработка подсистемы регистрации проекта;
- создание API, интеграция которого в код позволит произвести оповещение соответствующего персонала;
- разработка системы взаимодействия подсистем;
- тестирование;
- оценка качества работы системы.

Исследование в области разработки систем оповещения о сбоях в реальном времени представляет собой важный этап в повышении качества программного обеспечения и обеспечении стабильной работы информационных систем.

Структура и объём работы. Бакалаврская работа состоит из введения, 2 разделов, заключения, списка использованных источников и 5 приложений. Общий объем работы — 76 страниц, из них 57 страниц — основное содержание, включая 21 рисунков, цифровой носитель в качестве приложения, список использованных источников информации — 20 наименований.

Основное содержание работы

Описание проблемы. Важность разработки и внедрения системы оповещения о сбоях в коде в реальном времени трудно переоценить. Такая система не только помогает выявлять и реагировать на сбои мгновенно, но и способствует минимизации времени простоя, снижению рисков и обеспечивает более быстрое восстановление после возможных инцидентов. В результате этого, повышается общая устойчивость и надежность программных продуктов, что является критически важным аспектом в контексте современных информационных технологий.

Обзор существующих решений. В настоящее время не существует единого комплекса системы оповещений в реальном времени разработчиков о сбое в коде, который был бы универсален для всех платформ, языков программирования и типов оповещения.

В основном разработки в данной области направлены на предотвращения появления сбоев в коде, оценки: метрик, временных рядов, используемых ресурсов. Данный подход позволяет избежать многих сбоев, но не дает способа для быстрого реагирования на сбой. К таким системам можно отнести:

- Prometheus — это инструментарий для мониторинга систем и оповещения с открытым исходным кодом, первоначально созданный в SoundCloud.
- Redash — сервис для сбора и визуализации данных. Он может получать информацию из баз данных, таблиц, сервисов аналитики и других источников и представлять ее в виде таблиц, графиков, диаграмм и других визуализаций.

Архитектура системы. Система должна удволетворять следующим требованиям: микросервисная архитектура, наличие веб-интерфейса, возможность регистрации новых пользователей, различные способы оповещения, общение между сервисами через брокера сообщений.

Для создания требуемой системы была разработана следующая архитектура[1]:

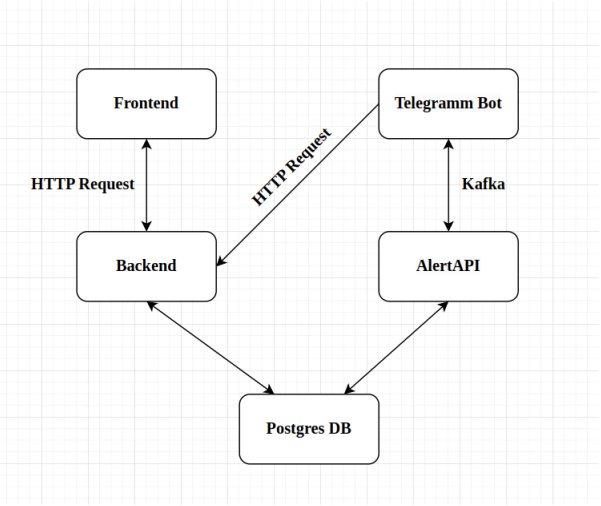

Рисунок 1 – Архитектура системы

Инструменты для решения задачи

Основным инструментом для разработки на Java является Spring.

Spring является комплексным фреймворком для разработки приложений. Его основные принципы включают инверсию управления (IoC) и внедрение зависимостей (DI), что способствует улучшению модульности и обеспечивает легкость тестирования.

В качестве базы данных используется Postgre SQL. Это мощная и полностью открытая реляционная система управления базами данных (RDBMS). С разнообразием возможностей и поддержкой сложных типов данных, PostgreSQL является одним из ведущих выборов для разработчиков и организаций, стремящихся к надежности и расширяемости баз данных.

Инструмент для обмена сообщениями. Apache Kafka — это распределенная система потоковой обработки и обмена сообщениями, предназначенная для эффективной передачи данных между различными компонентами системы. Разработанная Apache Software Foundation, эта платформа активно используется в современных высоконагруженных системах для обеспечения надежной и масштабируемой передачи данных.

Контейнеризация представляет собой технологию виртуализации на уровне приложения, позволяющую упаковывать и запускать приложения и их зависимости в изолированных контейнерах. Это обеспечивает консистентность выполнения приложений в различных окружениях и упрощает процессы развертывания, масштабирования и управления.

Docker — одна из популярных платформ для контейнеризации приложе-

ний.

Swagger — это инструмент для автоматической генерации интерактивной документации API. Он предоставляет возможность описывать структуру API, определять эндпоинты (конечные точки), параметры запросов, форматы передачи данных и другие аспекты в удобном для человека формате. Swagger поддерживает множество языков программирования и позволяет создавать документацию, которая остается актуальной в течение различных этапов разработки приложения.

Разработка программного комплекса.

В данном разделе рассматриваются основные этапы разработки системы оповещения о сбоях в коде в реальном времени.

База данных.В качестве базы данных используется Postgre SQL.

В конестексте данной системы существуют две сущности: люди и проекты.

У человека в системе должны быть следующие поля: id, логин, пароль, емейл, телефонный номер и ссылка на Telegram.

У проекта в системе должны быть следующие поля: id, название, описание, ссылка на чат в Telegram.

Схема, приведенная на рисунке 2, описывает требуемую базу данных:

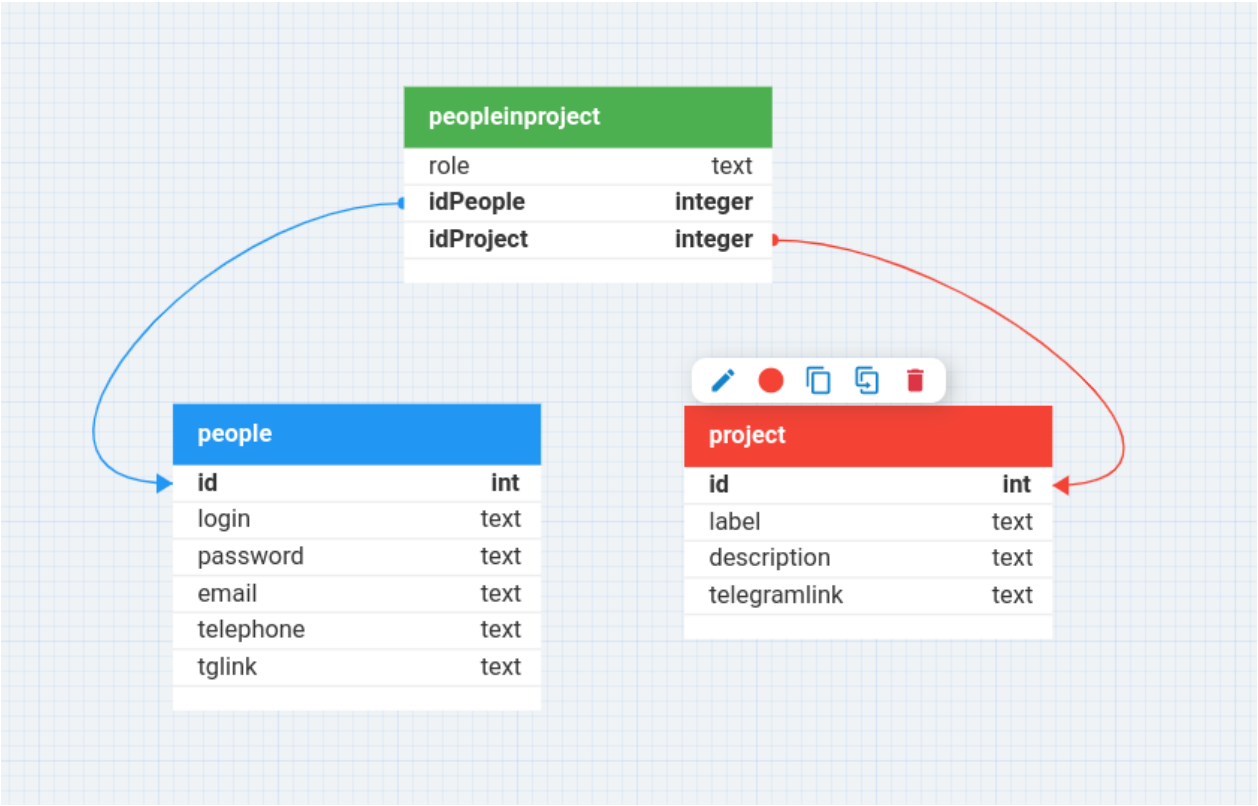

Рисунок 2 – Схема базы данных

Работа с базой данных внутри приложения. При работе с базами данных возникает выбор между написанием запросов на SQL или использованием ORM. С одной стороны, SQL является мощным инструментом для работы с данными, позволяя разработчикам выражать сложные запросы и манипулировать базами данных напрямую. С другой стороны, использование SQL требует глубокого понимания структуры базы данных и языка запросов [2].

Для реализации требуемой базы данных, нужно описать следующие сущности: People, Project, PeopleInProject, PeopleInProjectId. Первые три сущности соответсвуют таблицам в базе данных [3]. Сущность PeopleInProjectId используется для создания составного ключа, так как в таблице PeopleInProject, первичны ключе это: idPeople и idProject.

Серверная часть системы. В данном разделе работы описаны шаги для создания серверной части сайта:

— создание модуля аутентификации [4];
— создание модуля для получения данных с backend, по средствам реализации контроллеров;

Клиентская часть системы. В данном разделе работы будет описан процесс создания пользовательской части сайта для сервиса AlertAPI.

С помощью сайта у пользователя должны быть следующие возможности:

1. Узнать данные о проектах;
2. Получить развернутые данные о каждом проекте;
3. Создавать проекты и выходить из них;
4. Получать данные о своем аккаунте;
5. Редактировать данные о своем аккаунте;

Для реализации функционала всех возможностей на пользовательской стороне был разработан код на html, css, js.

Настройка CORS. В данном разделе работы рассматривается важный аспект веб-разработки — кросс-доменные запросы (CORS, Cross-Origin Resource Sharing) и способ их настройки в приложениях на Spring Boot. CORS является механизмом, который позволяет веб-приложениям расширять безопасность на уровне браузера и контролировать доступ к ресурсам на других доменах.

Аннотация @CrossOrigin в Spring Boot позволяет управлять CORS (Cross-Origin Resource Sharing) для конкретного метода контроллера или контроллера в целом.

В данной системе аннотация @CrossOrigin добавляется на уровне контроллера:

```
@CrossOrigin(origins = "http://127.0.0.1:5500", allowCredentials = "true")
@RestController @RequestMapping("/auth") public class AuthController { ... }
```

— Параметр origins — Указывает разрешенные источники запросов. В данном случае указан источник http://127.0.0.1:5500;

— Параметр allowCredentials — Указывает, разрешено ли передавать учетные данные в запросах CORS. В данном случае установлено значение true.

Разработка микросервиса AlertAPI. Для реализации микросервиса по оповещению о сбоях в коде, достаточно реализовать один контроллер. Данный контроллер будет принимать запросы по адресам /alert/byRole/role и /alert/byId/id и соответсвенно оповещать пользователя с указанным id или группу пользователей с указанной ролью.

Telegram бот. Для создания Telegram бота используется библиотека aiogram. Aiogram — это асинхронная библиотека Python для создания ботов Telegram. Она предоставляет удобный интерфейс для взаимодействия с API Telegram и управления ботами через асинхронные вызовы. Версия библиотеки используемая в проекте 2.14.

Данный код создает бота на основе токена. Токен выдается Telegram для управления ботом. Затем происходит создание Dispatcher. Dispatcher в библиотеке aiogram представляет собой центральное управление для обработки всех входящих сообщений и событий от Telegram API.

Бот просит пользователя авторизовать свой проект. После того как пользователь написал свои данные, происходит выполнение функции login.

Данные считываются из сообщения и конструируется POST запрос на Backend. Затем происходит проверка успешного выполнения запроса.

Если все успешно, чат сохраняется в базе данных для данного проекта.

Сообщение пользователя удаляется, и запускается функция on_kafka_message, которая прослушивает сообщения от Kafka и присылает их в чат.

Брокер сообщений. Apache Kafka — это распределенная система потоковой обработки данных и сообщений, предназначенная для управления и обработки потоков данных в реальном времени.

Для того, чтобы Kafka работала, нужно запустить ZooKeeper service и Kafka broker service.

В данном проекте в роли продюсера выступает Alert API, а точнее класс TelegramAlert [5].

В данной работе для работы с Kafka в python используется библиотека aiokafka. При использовании стандартной библиотеке бот бы не мог работать параллельно с Kafka.

На основе объекта properties создается Producer, создается объект сообщения TelegramAlertDTO, и отсылается с помощью функции send объекта producer в тему topic, со значением ключа key, и сообщением value.

Документирование микросервисов. В данном разделе представлено описание API "Send Alert API"с использованием Swagger спецификации в формате OpenAPI 3.0.0. Swagger спецификация используется для описания структуры и функциональности веб-сервисов и API. Она позволяет разработчикам автоматически создавать, документировать и тестировать API, обеспечивая понятное и удобное взаимодействие между клиентами и серверами.

Контенеризация системы. В данном разделе работы, описаны шаги для контенеризации всей систимы, которая описана в предыдущих разделах.

Для каждой подсистемы создается dcokerfile. Dockerfile — это текстовый файл, который содержит инструкции по сборке образа Docker. В Dockerfile определяются все необходимые шаги для создания контейнера, включая базовый образ, установку зависимостей, копирование файлов, настройку окружения и так далее.

В данной системе можно выделить следующие подсистемы для которых нужно описать dockerfile:

- Frontend;
- Backend;
- Alert API;
- Telegram бот;
- Kafka;
- База данных Postgres;

Демонстрация работы системы. В данном разделе работы представлена демонстрация работы разработанной системы, позволяющая рассмотреть ее функциональность и возможности в действии. Во-первых, здесь представленны

основные возможности пользовательского интерфейса и его взаимодействие с пользователем, во вторых, типичные сценарии использования системы.

Цель данного раздела — предоставить полное представление о функциональности и преимуществах разработанной системы, а также продемонстрировать ее потенциал.

ЗАКЛЮЧЕНИЕ

Была разработана система оповещения о сбоях в коде в реальном времени, способная интегрироваться в код на любом языке программирования, со следующим функционалом:

- аутентификация и авторизация на сайте проекта;
- создание и редактирование проктов;
- Swagger API для быстрой интеграции пользователя с его приложениями;
- отправка уведомлений о сбоях в проекте по SMS, email, Telegram;

В данной работе был рассмотрен процесс построения микросервисного приложения, составления базы данных, технологии ORM, позволяющие работать с базами данных, был изучен фреймворк Spring с его основными компонентами: Spring Data, Spring Boot, Spring Core, Spring Security. Был изучен процесс настройки CORS, динамическое создание html объектов, контейнеризация системы, написание асинхронных приложений, в частности — чат бота Telegram совместно с Kafka. Также было изучено средство документирования микросервисов, такое как Swagger. Помимо этого были освоены библиотеки для работы с отправкой смс и email в Java.

Были достигнуты все цели и выполнены поставленные задачи.

Система имеет следующие преимущества:

- легкая интеграция с любым языком программирования;
- прямая интеграция в код;
- отсутвие привязки к метрикам функционирования системы;
- мгновенноя отправка уведомления о сбое;
- возможность выбора, куда именно должно предти оповещение;
- открытый исходный код системы;

Данная система может использоваться людьми, для которых очень важно моментальное устранение неполадок в системе, людьми, которые хотят сделать систему пересылки сообщений между различными платформами, а также людям, которые занимаются анализом дефектов в коде.