логи с уровнем INFO

Уровень INFO предназначен для предоставления информации о нормальных событиях системы, которые помогают отслеживать её работу без необходимости вмешательства со стороны оператора. Логирование на уровне INFO должно включать события, которые дают представление об общем состоянии системы, успешном выполнении задач и важных переходах между состояниями.

Логи уровня INFO необходимо собирать для каждой ключевой системы:

### **Онлайн-магазин**

* **Создание заказа**
  + Идентификатор заказа, время создания, пользователь, список товаров.
* **Обновление заказа**
  + Идентификатор заказа, статус, время обновления.
* **3D-модель загрузка и создание**
  + Время загрузки, пользователь, параметры модели (например, количество полигонов).

### **CRM**

* **Создание и обновление клиента**
  + Идентификатор клиента, тип обновления, время.
* **Получение и обработка заказов**
  + Входящие сообщения из очереди, идентификаторы заказов.

### **MES**

* **Запуск и завершение производственного задания**
  + Идентификатор задания, время начала и окончания, статус.
* **Обработка стоимости изделия**
  + Идентификатор запроса, время начала и завершения, сложность модели.
* **Проблемы с загрузкой страницы**
  + Время и длительность загрузки, количество отображаемых заказов, примененные фильтры.

### **Общие:**

* **Интерфейсы API**
  + Входящие и исходящие запросы, время и статус обработки.
* **Взаимодействие через RabbitMQ**
  + Отправленные и полученные сообщения, идентификатор сообщения, время обработки.

### **Обозначение необходимых мест для логирования на схеме:**

1. **Интернет-магазин (Shop API, Shop DB)**:
   * Логирование всех операций по заказам и 3D-моделям.
2. **CRM (CRM API, CRM DB)**:
   * Вся логика по клиентам и обработке заказов.
3. **MES (MES API, MES DB)**:
   * Логирование запуска и завершения производственных заданий и оценки стоимости.
4. **Очереди сообщений (Message Queue/RabbitMQ)**:
   * Логирование всех операций по отправке и приёму сообщений.
5. **Внутренний API и внешние интерфейсы**:
   * Логи всех взаимодействий через API.

Напишите, будете ли вы использовать другие уровни логирования и при каких обстоятельствах

### **TRACE**

* **Использование**: Для самого подробного уровня логирования.
* **Когда**: Для отслеживания каждого шага выполнения, например, при сложных интеграциях и отладке.

### **DEBUG**

* **Использование**: Для подробной информации, полезной в процессе разработки и отладки.
* **Когда**: При разработке новых функций или анализе сложных проблем.

### **INFO**

* **Использование**: Для общей информации о нормальной работе системы.
* **Когда**: Логирование стандартных операций, таких как создание и обновление заказов.

### **WARN**

* **Использование**: Для обозначения потенциальных проблем, которые не требуют немедленного вмешательства.
* **Когда**: Уведомления о снижении производительности или необычном поведении.

### **ERROR**

* **Использование**: Для записи ошибок, влияющих на функциональность системы.
* **Когда**: Сбои операций, нарушения функциональности, например, сбои API.

### **FATAL**

* **Использование**: Для критических ошибок, требующих немедленного вмешательства.
* **Когда**: Системные сбои, критические проблемы, ведущие к остановке сервисов.

Мотивация

Добавление логирования в систему позволяет обнаруживать, анализировать и решать технические проблемы, обеспечивая более высокую стабильность и производительность системы. Это также способствует улучшению пользовательского опыта и повышению доверия клиентов.

### **Преимущества для компании**

1. **Улучшенная диагностика проблем**
   * Быстрое выявление и устранение ошибок.
2. **Повышенная надежность**
   * Превентивное обнаружение и предотвращение сбоев.
3. **Улучшенный пользовательский опыт**
   * Меньшее количество сбоев и ускоренное решение проблем повышают удовлетворенность клиентов.
4. **Аналитика и оптимизация**
   * Сбор данных для анализа производительности и пользовательского поведения.

### **Технические и бизнес-метрики**

1. **Время отклика системы**
   * Логирование позволяет определить и устранить узкие места, улучшая скорость отклика приложений.
2. **Доступность системы**
   * Снижение времени простоя путем быстрого обнаружения и устранения инцидентов.
3. **Количество ошибок и сбоев**
   * Снижение числа критических ошибок благодаря своевременному их обнаружению и обработке.
4. **Пользовательская активность**
   * Анализ поведения пользователей для улучшения интерфейса и функциональности.
5. **Удовлетворенность клиентов**
   * Обратная связь от клиентов при улучшении стабильности и быстродействия приложения.

### **Приоритетные системы для логирования и трейсинга**

1. **CRM**
   * **Почему**: Центральная система для управления отношениями с клиентами. Проблемы в ней напрямую влияют на взаимодействие с клиентами и удовлетворенность.
2. **MES**
   * **Почему**: Критично для производственных процессов. Любые задержки или ошибки могут значительно повлиять на выполнение заказов.
3. **Онлайн-магазин**
   * **Почему**: Является лицом компании для клиентов, и любые проблемы могут привести к снижению доверия и потере продаж.

### 

Начальная настройка логирования и трейсинга в этих системах позволит улучшить видимость, диагностику и быстроту реагирования на проблемы, что положительно скажется как на технических, так и на бизнес-метриках компании.

Предлагаемое решение

### **Реализация логирования**

#### **Технологии**

1. **Логгирование в приложениях**
   * **Log4j/Logback** для Java (онлайн-магазин, CRM).
   * **NLog/Serilog** для C# (MES).
2. **Система сбора и агрегации логов**
   * **ELK Stack (Elasticsearch, Logstash, Kibana)** для сбора и визуализации логов.
   * **Grafana** для мониторинга, если требуется более продвинутый анализ.
   * **Fluentd** как альтернативный лог-агрегатор.
3. **Облачное хранилище**
   * **S3 или аналог (в Yandex Cloud)** для долговременного хранения.
   * **Yandex Cloud Logging** как встроенное решение облачной логгинга.

#### **Компоненты и новые связи**

1. **Логирование в каждом приложении (онлайн-магазин, CRM, MES)**
   * Внедрение библиотек логирования.
2. **Лог-агрегатор (Logstash/Fluentd)**
   * Интеграция с приложениями для передачи логов в централизованное хранилище.
3. **Elasticsearch и Kibana**
   * Визуализация и поиск по логам.
4. **Облачное хранилище**
   * Резервное хранение и архивирование старых логов.

### **Политика безопасности**

1. **Шифрование**
   * Шифрование логов при передаче (HTTPS) .
2. **Доступ**
   * Ограниченный доступ через ролевую модель (IAM), доступ только определенным сотрудникам (администраторы, DevOps).
   * Логи с чувствительной информацией должны иметь маскирование данных — исключение или хеширование PII (персонально идентифицируемой информации).

### **Политика хранения**

1. **Индексация**
   * Разделение индексов по системам (онлайн-магазин, CRM, MES).
   * Отдельные индексы для разных уровней логирования.
2. **Срок хранения**
   * **Горячие данные** (активные логи) — 30 дней.
   * **Теплые данные** — архивация на 3 месяца.
   * **Холодные данные** — долговременное хранение до 1 года на S3.
3. **Размер и ротация**
   * Ограничение размера индекса, автоматическая ротация и удаление старых логов согласно политике.

мероприятия

### **Необходимые мероприятия для превращения системы сбора логов в систему анализа логов**

1. **Настройка алертинга**
   * **Инструменты**: Использовать мониторинг и алертинг с помощью **Elasticsearch Watcher** или **Grafana Alerts**.
   * **Настройки**:
     + Уведомления о сбоях в работе API или значительном увеличении ошибок (ERROR/FATAL).
     + Алерты на превышение времени обработки запросов, что может указывать на узкие места.
     + Уведомления об увеличении нагрузки (например, резкий всплеск создания заказов).
2. **Поиск аномалий**
   * **Инструменты**: Внедрить **Elastic Machine Learning** для автоматического поиска аномалий.
   * **Примеры**:
     + Резкое увеличение количества заказов, что может свидетельствовать о DDoS-атаке.
     + Необычные паттерны в поведении пользователей, такие как чрезмерное количество логинов.
3. **Дашборды и визуализация**
   * Создать детализированные дашборды в **Kibana**:
     + Обзор по количеству запросов на единицу времени.
     + Метрики производительности (время обработки заказа, загрузка страниц).
     + История и тренды ошибок.
4. **Интеграция с другими системами**
   * Связь с системами безопасности для немедленного реагирования на выявленные угрозы.
   * Использование SIEM-систем для комплексного анализа безопасности.
5. **Оптимизация и повышение эффективности**
   * Регулярный анализ ресурсов потребления логов для оптимизации хранилищ и индексов.
   * Настройка архивирования и удаления устаревших данных для поддержания эффективности системы.

### 

Реализация этих мер позволит эффективно анализировать логи и обеспечивать своевременное реагирование на аномалии и инциденты, способствующие повышению устойчивости и безопасности системы.