Университет ИТМО

Факультет программной инженерии и компьютерной техники

«Бизнес-логика программных систем»

Лабораторная работа №4

Вариант 331572

Студенты:
Ляо Ихун

Группа:
P33131

Преподаватель:
Кулинич Ярослав Вадимович

Санкт-Петербург 2023

Задание

Доработать приложение из лабораторной работы #2, реализовав в нём асинхронное выполнение задач с распределением бизнес-логики между несколькими вычислительными узлами и выполнением периодических операций с использованием планировщика задач.

Требования к реализации асинхронной обработки:

1. Перед выполнением работы неободимо согласовать с преподавателем набор прецедентов, в реализации которых целесообразно использование асинхронного распределённого выполнения задач. Если таких прецедентов использования в имеющейся бизнес-процесса нет, нужно согласовать реализацию новых прецедентов, доработав таким образом модель бизнес-процесса из лабораторной работы #1.
2. Асинхронное выполнение задач должно использовать модель доставки "подписка".
3. В качестве провайдера сервиса асинхронного обмена сообщениями необходимо использовать сервис подписки на базе Apache Kafka + ZooKeeper.
4. Для отправки сообщений необходимо использовать Kafka Producer API.
5. Для получения сообщений необходимо использовать API KafkaConsumer (org.apache.kafka.clients.consumer.KafkaConsumer).

Требования к реализации распределённой обработки:

1. Обработка сообщений должна осуществляться на двух независимых друг от друга узлах сервера приложений.
2. Если логика сценария распределённой обработки предполагает транзакционность выполняемых операций,

они должны быть включены в состав распределённой транзакции.

Требования к реализации запуска периодических задач по расписанию:

1. Согласовать с преподавателем прецедент или прецеденты, в рамках которых выглядит целесообразным использовать планировщик задач. Если такие прецеденты отсутствуют -- согласовать с преподавателем новые и добавить их в модель автоматизируемого бизнес-процесса.
2. Реализовать утверждённые прецеденты с использованием планировщика задач Quartz.

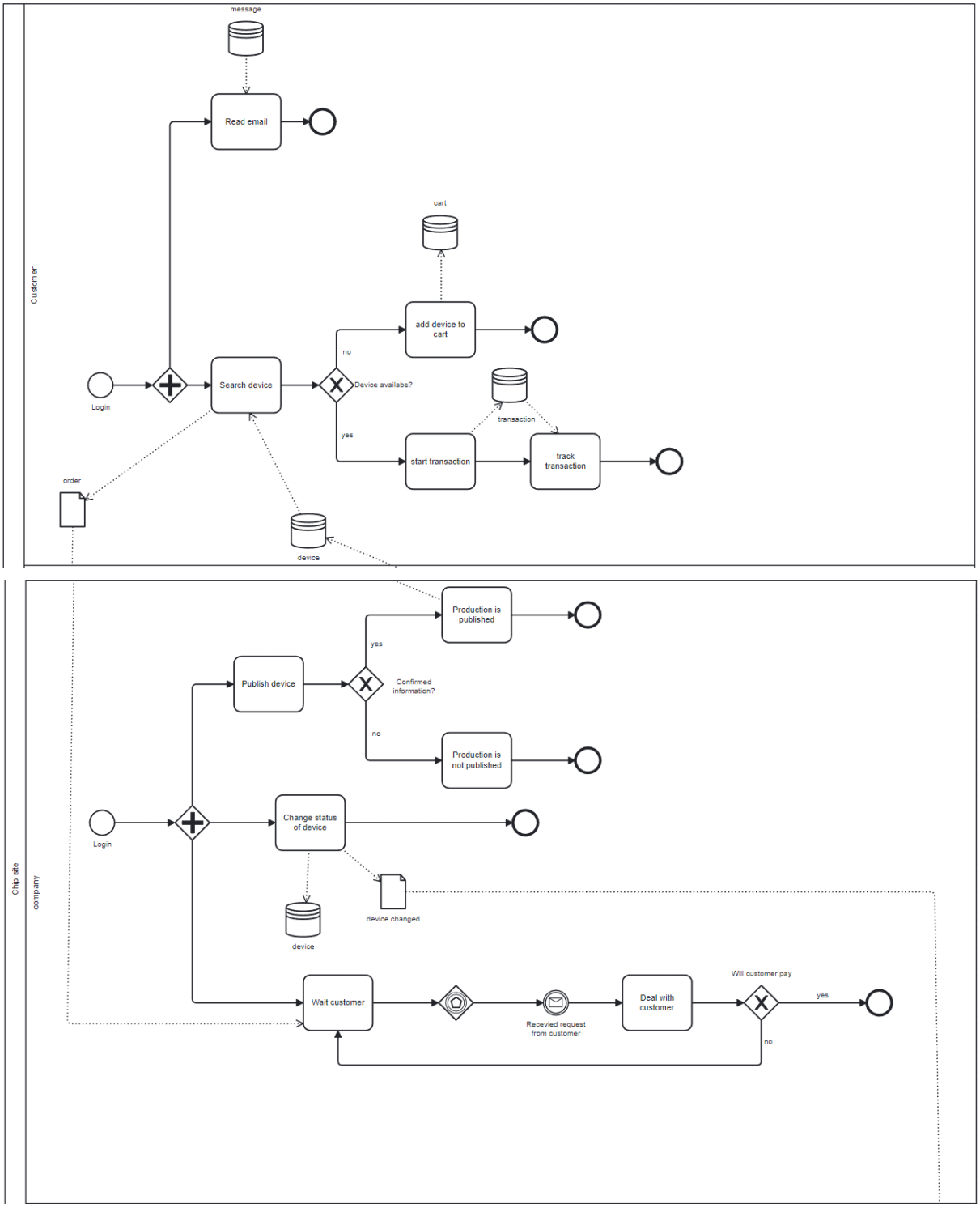
Правила выполнения работы:

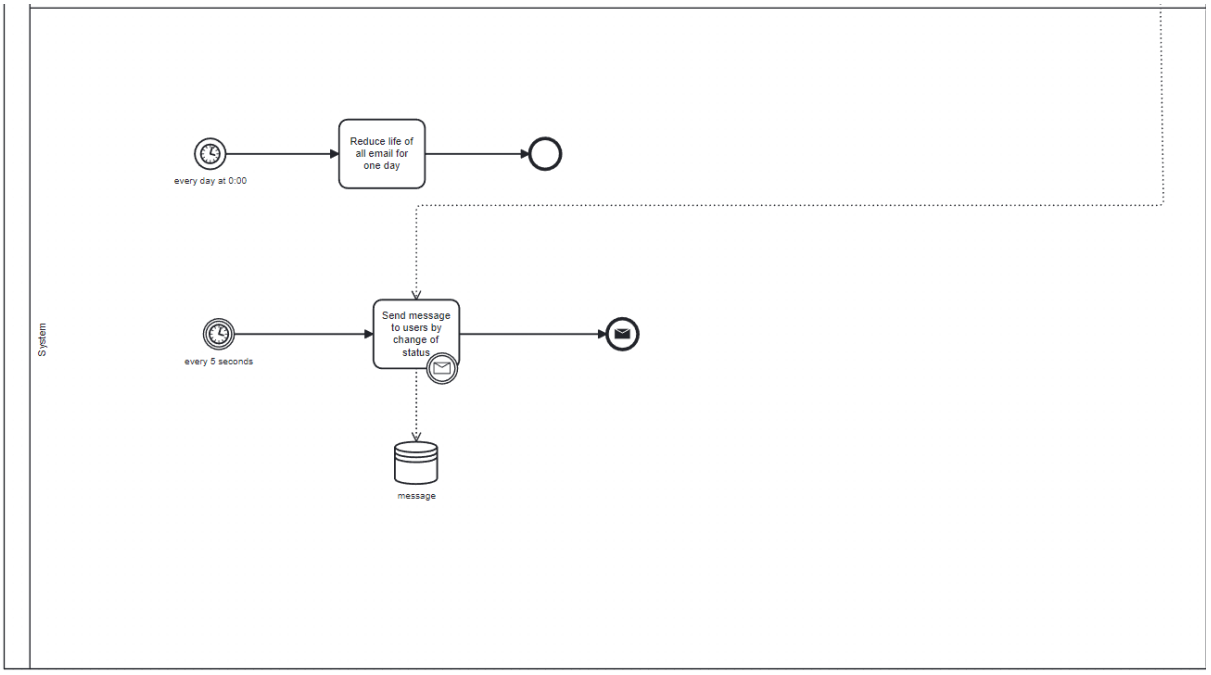
1. Все изменения, внесённые в реализуемый бизнес-процесс, должны быть учтены в описывающей его модели, REST API и наборе скриптов для тестирования публичных интерфейсов модуля.
2. Доработанное приложение необходимо либо развернуть на сервере helios, либо продемонстрировать его работоспособность на собственной инфраструктуре обучающегося.

Содержание отчёта:

1. Текст задания.
2. Модель потока управления для автоматизируемого бизнес-процесса со всеми внесёнными изменениями.
3. UML-диаграммы классов и пакетов разработанного приложения.
4. Спецификация REST API для всех публичных интерфейсов разработанного приложения.
5. Исходный код системы или ссылка на репозиторий с исходным кодом.
6. Выводы по работе.

Описание


message
Read email
Customer
cart
add device to cart
no
Login
Search device
Device availabe?
transaction
yes
start transaction
track transaction
order
device
Production is published
yes
Publish device
Confirmed information?
no
Production is not published
Chip site company
Login
Change status of device
device changed
device
Wait customer
Recevied request from customer
Deal with customer
Will customer pay
yes
no



Проект

[Ссылка к проекту]

Результат

В результате получается что на приложения запускаюься с kafka и периодически выполняются задачи с Quartz.

Kafka

```
2023-06-21 23:52:08.163  INFO 33800 --- [           main] o.a.kafka.common.utils.AppInfoParser     : Kafka version: 3.0.0
2023-06-21 23:52:08.164  INFO 33800 --- [           main] o.a.kafka.common.utils.AppInfoParser     : Kafka commitId: 8cb0a5e9d3441962
2023-06-21 23:52:08.164  INFO 33800 --- [           main] o.a.kafka.common.utils.AppInfoParser     : Kafka startTimeMs: 1687380728162
2023-06-21 23:52:08.164  INFO 33800 --- [           main] o.a.k.clients.consumer.KafkaConsumer     : [Consumer clientId=consumer-group2-1, groupId=group2] Subscribed to topi
2023-06-21 23:52:08.215  INFO 33800 --- [           main] o.a.k.clients.consumer.ConsumerConfig    : ConsumerConfig values:
    allow.auto.create.topics = true
    auto.commit.interval.ms = 5000
    auto.offset.reset = latest
    bootstrap.servers = [localhost:9092]
    check.crcs = true
    client.dns.lookup = use_all_dns_ips
    client.id = consumer-group1-2
    client.rack =
    connections.max.idle.ms = 540000
    default.api.timeout.ms = 60000
    enable.auto.commit = true
    exclude.internal.topics = true
    fetch.max.bytes = 52428800
    fetch.max.wait.ms = 500
    fetch.min.bytes = 1
    group.id = group1
    group.instance.id = null
    heartbeat.interval.ms = 3000
    interceptor.classes = []
    internal.leave.group.on.close = true
    internal.throw.on.fetch.stable.offset.unsupported = false
    isolation.level = read_uncommitted
```

Quartz

```
2023-06-21 23:52:09.426  INFO 33800 --- [           main] o.s.s.quartz.SchedulerFactoryBean        : Starting Quartz Scheduler now
2023-06-21 23:52:09.426  INFO 33800 --- [           main] org.quartz.core.QuartzScheduler          : Scheduler quartzScheduler_$_NON_CLUSTERED started.
2023-06-21 23:52:09.432  INFO 33800 --- [           main] edu.itmo.blps.DemoApplication            : Started DemoApplication in 4.448 seconds (JVM running for 5.041)
Hibernate: select message0_.id as id1_4_, message0_.customer as customer2_4_, message0_.life as life3_4_, message0_.text as text4_4_ from message message0_
2023-06-21 23:52:10.050  WARN 33800 --- [eduler_Worker-1] bitronix.tm.twopc.Preparer               : executing transaction with 0 enlisted resource
2023-06-21 23:52:11.495  INFO 33800 --- [ionShutdownHook] org.quartz.core.QuartzScheduler          : Scheduler quartzScheduler_$_NON_CLUSTERED paused.
2023-06-21 23:52:11.523  INFO 33800 --- [ionShutdownHook] o.s.s.quartz.SchedulerFactoryBean        : Shutting down Quartz Scheduler
2023-06-21 23:52:11.523  INFO 33800 --- [ionShutdownHook] org.quartz.core.QuartzScheduler          : Scheduler quartzScheduler_$_NON_CLUSTERED shutting down.
2023-06-21 23:52:11.523  INFO 33800 --- [ionShutdownHook] org.quartz.core.QuartzScheduler          : Scheduler quartzScheduler_$_NON_CLUSTERED paused.
2023-06-21 23:52:11.523  INFO 33800 --- [ionShutdownHook] org.quartz.core.QuartzScheduler          : Scheduler quartzScheduler_$_NON_CLUSTERED shutdown complete.
```

Вывод

В ходе выполнении работы, при помощи kafka добавили producer сообщения, кто отправляет сообщения в группы consumer, в результат предыдующей лабы, и создали второй север, где есть две группы consumers, кто получает сообщения от producer и оформляет сообщение к пользователю приложения. Создали две периодической задачи: управление жизнью сообщения пользователи и сбор сообщения от producer. Для реализации периодических задач применяется Quartz. Изучены применения kafka и Quartz в рамке springboot.