1. Создан пилотный проект — спрингбут приложение: wa\_monitoring.jar

2. Созданы java сущности:

- AppUser.java

- AuthKey.java

- SubscriptionPhone.java

- VisitNote.java

3. Разработаны внутренние микросервисы приложения:

- StandardPersistenceService.java — сохранение сущностей в БД

- VisitNoteReactor.java — реактивная обработка VisitNote

- WaMonitoringRest.java

- LoginReactor.java

- SecurityRest.java

- MonitoringRSocketService.java

- AuthenticationManager.java

4. Указанные сущности замаплены на соответствующие объекты БД Oracle (jdbc:oracle:thin:@//109.234.37.16/ORCLPDB1)

Разработы следующие http эндпоинты:

- /api/createAppUser

- /api/createAuthKey

- /api/createSubscriptionPhone

- /api/createVisitNote

- /api/createHiLoadVisitNote

Назначение каждого эндпоинта — заведение соответствующей сущности в БД Oracle

/api/createHiLoadVisitNote — эндпоинт предназначен для массового сохранения сущности VisitNote батчем. Эндпоинт принимает данные для заведения сущности VisitNote, и формирует responce клиенту (3-5ms), принятые сущности складывает в коллекцию и по расписанию (раз в ХХХ ms) батчем сохраняет в БД.

- Каждый из эндпоинтов требует наличия в заголовка Authorization и указанного в нем токена

Bearer \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

- Для режима разработки и тестирования можно отключить проверку токена и авторизацию пользователя, приславшего запрос

- Токен может быть получен при авторизации клиента в ответе на поинт /api/login

5. Разработаны конфигурции спрингбут приложения, которые можно отключать и включать при помощи настроечных параметров (@Conditional, @ConditionalOnProperty)

6. Для включения\отключения вставки и апдейта хибернейтом батчами, существуют следующие настроечные параметры (application.yml)

hibernate:

generate\_statistics: true

order\_inserts: true

order\_updates: true

jdbc:

batch\_size: 500

Параметр generate\_statistics позволяет мониторить через лог статиситку формирования батчей, например:

2021-02-23 17:05:49.170 INFO 16910 [ scheduling-1] i.StatisticalLoggingSessionEventListener : Session Metrics {

47493439 nanoseconds spent acquiring 1 JDBC connections;

0 nanoseconds spent releasing 0 JDBC connections;

44596574 nanoseconds spent preparing 599 JDBC statements;

28611634372 nanoseconds spent executing 598 JDBC statements;

157084344 nanoseconds spent executing 2 JDBC batches;

0 nanoseconds spent performing 0 L2C puts;

0 nanoseconds spent performing 0 L2C hits;

0 nanoseconds spent performing 0 L2C misses;

236186196 nanoseconds spent executing 1 flushes (flushing a total of 598 entities and 0 collections);

0 nanoseconds spent executing 0 partial-flushes (flushing a total of 0 entities and 0 collections)

Пример генерации запросов хибернейтом из лога:

Hibernate: select seq\_visitnote.nextval from dual

Hibernate: insert into visitNote (ADD\_TIME, IS\_ONLINE, phone\_id, id) values (?, ?, ?, ?)

7. Разработаны JUnit тесты для каждой из вышеуказанных сущностей:

- AppUserTests.java

- AuthKeyTests.java

- SubscriptionPhoneTests.java

- VisitNoteTests.java

- каждый файл содержит набор тестов для конкретной сущности,

- каждый из тестов может быть повторен нужное количество раз (@RepeatedTest(1000)) или отключен в режиме разработки

- каждый тест может быть запущен индивидуально в режиме разработки,

- также все тесты могут быть запущены последовательно во время сборки версии (maven clean install), также сборка версии может быть запущена без запусков тестов (-DskipTests=true)

- каждый из тестов через соотвествующий эндпоинт шлет сообщение с данными для создания или обновления конкретной сущности

8. Проект с исходниками спринг-бут приложения (wa\_monitoring) прикреплен к письму

9. Кусок рабочего лога (log4j2) также прикреплен к письму