## Кейс №3

### 📚 Легенда

Помимо построения расписания рекомендуемых визитов, наш новый “р*ис*ковый” ИИ также стал основой для ещё одной мега фичи – выдачи советов по уходу за животными каждому владельцу персонально. Эта функция перебирает всех владельцев животных, зарегистрированных в клинике, ~~якобы~~ анализирует их собственные предпочтения (пол, возраст, характер занятости), а затем с их учётом ~~якобы~~ прорабатывает медицинскую карту каждого животного и формирует советы по уходу за ним.

На первых порах фича запускается лишь раз при старте приложения и формирует советы сразу по всем владельцам и всем питомцам.

### ⚙️ Техническая реализация

Вся фича реализована в классе (сервисе) `CareTipsProvider`, который активируется свойством `enable-care-tips` через аргумент командной строки или настройки приложения.

Входной точкой является метод `composeCareTips`, который запускается Spring’ом как слушатель события `ApplicationReadyEvent`.

Поскольку анализ каждого владельца и каждого животного – это потенциально времязатратные операции, они выполняются параллельно, причём как для владельцев, так и для их питомцев.

В качестве средства распараллеливания используется отдельно заведенный пул `CareThreadPoolConfig#careTipsThreadPool` с фиксированным числом потоков, которое по умолчанию равно количеству ядер процессора, так как делать меньше не выгодно, а больше – нет смысла. Очередь у пула потоков есть, и она не ограничена, чтобы все попадающие в него задачи рано или поздно начинали исполнение (см. javadoc на `Executors#newFixedThreadPool(int)`).

Тот факт, что задачи формирования советов могут занимать разное время, учитывается вызовом метода `ExecutorService.invokeAll()`, который возвращает управление лишь после того, как *все* переданные ему задачи завершились.

Метод `CareTipsProvider#composeCareTips` достаёт из БД всех владельцев питомцев и для каждого из них создаёт экземпляр задачи `OwnerCareTask`, а затем все получившиеся задачи разом отправляет на исполнение методом `ExecutorService#invokeAll` указанного выше пула потоков. Метод возвращает управление после завершения всех задач и выдаёт результаты в виде списка. Список обходится последовательно, и каждый элемент (получившийся набор советов) выводится в лог приложения.

Задача `OwnerCareTask` работает в контексте одного владельца и первым делом загружает из БД всех его питомцев. Затем для каждого питомца она создаёт экземпляр задачи `PetCareTask` и все получившиеся задачи разом отправляет на исполнение методом `invokeAll` того же самого пула потоков. Полученные результаты (советы по отдельным питомцам) объединяются в один (потенциально многострочный) текст и возвращаются в качестве результата всей задачи `OwnerCareTask`.

Задача `PetCareTask` отвечает за фактический подбор советов в разрезе одного питомца и на данном этапе возвращает всегда один и тот же константный совет в виде строки.

Таким образом, общая логика такова, что главный поток ответвляет несколько потоков с задачами `OwnerCareTask` для владельцев, а те, в свою очередь, ответвляют по несколько потоков с задачами `PetCareTask` для питомцев. Затем все результаты собираются воедино и выводятся в лог одним структурированным сообщением. Такая схема позволяет максимально утилизировать имеющиеся вычислительные мощности и как можно раньше вернуть управление, чтобы приложение могло приступить к своей основной работе.

### 🐞 Проблема

В некоторых случаях построение советов при запуске приложения зависает. В логах при этом видно, что оно хотя бы началось:

```
13:15:58.473 DEBUG CareTipsProvider - Proposing care tips for 10 owners...
13:15:58.474 DEBUG OwnerCareTask - Proposing care tips for 1 pet(s) of owner
George...
13:15:58.475 DEBUG OwnerCareTask - Proposing care tips for 1 pet(s) of owner
Harold...
13:15:58.475 DEBUG OwnerCareTask - Proposing care tips for 1 pet(s) of owner
Betty...
13:15:58.475 DEBUG OwnerCareTask - Proposing care tips for 2 pet(s) of owner
Eduardo...
```

, однако больше никаких сообщений не выводится и ничего не происходит бесконечно долго. Само приложение (в частности, его веб-сервер) при этом работает.

Предположительно, проблема как-то связана с аппаратной частью машины, так как стабильно воспроизводится на одних компьютерах и вообще никак не проявляет себя на других, хотя операционные системы одинаковы.

Дополнительное наблюдение: после возникновения проблемы штатный останов приложения становится невозможен, приходится “убивать” процесс.

### 👣 Шаги воспроизведения

1. Запустить приложение с аргументом `--enable-care-tips`

***Ожидаемый результат***: через несколько секунд в лог выводится INFO-сообщение вида:

```
CareTipsProvider - General care tips from Spring PetClinic:
Dear George, here is your pet care tip(s):
  - Your Leo should sleep more
...
```

***Фактический результат***: в логе нет таких сообщений.

*Примечание*. Если проблема не воспроизводится, можно попробовать сэмулировать менее производительную машину, передав JVM-процессу опцию `-XX:ActiveProcessorCount=` со значением, заметно уступающим фактическому числу ядер процессора, например, 4. При запуске в Docker вместо этой опции нужно указать опцию `--cpus 4` для команды `docker run`.