Техническое задание
на разработку системы сбора данных с сайта-витрины (маркетплейса)

1. Необходимо разработать систему сбора данных с витрины с последующим формированием дампа полученных данных в соответствии со следующими требованиями.

№	Функция / свойства	Требования к реализации
1	Наименование сайта-витрины	Сбор данных нужно выполнить с _____.
2	Функциональные требования	Система должна обеспечить полный сбор данных (перечень указан ниже) с сайта-витрины в течение 1 суток. Данные должны быть собраны в полном объеме. Сбор данных должен производится параллельными процессами с возможностью горизонтального масштабирования за счет добавления нового числа нод в кластер. Собранные данные должны быть сформированы в 4 дампа в формате JSON: - products; - merchant; - position; - review. Полученные дампы должны быть автоматически выгружаться на сервер-хранилище заказчика.
3	Состав собираемых данных	**Product # данные получаются на странице карточки продукта** - id # идентификатор продукта на витрине - merchant - id # идентификатор магазина на витрине; совпадает с Merchant.id для связи данных Product -> Merchant - category - id # идентификатор категории на витрине; продукт отнесён к одной из категории в каталоге витрины - path # идентификаторы категорий; путь от корня каталога до конечной категории (id выше), перечислены категории 1-го, 2-го, 3-го и т.д. уровней - name # наименование продукта; не путать с описанием, которое более объемное - ?created # дата создания продукта; не все витрины публикуют поле - ?image # адрес по-умолчанию избражения продукта; обычно используется в лентах - images # полный список адресов изображений; отображается уже на карточке продукта - rating # рейтинг продукта - ?ratingCount # число оценок; не путать с числом отзывов - ?ratingDistr # распределение оценок по баллам; сумма чисел оценок каждого балла обычно совпадает с общим их числом ratingCount - ?reviewsCount # число отзывов на продукт; не путать с числом оценок - ?likesCount # число лайков продукта; может интерпретироваться как "понравилея", "желаемое", "рекомендуемое" и т.п.

		как "понравился", "желаемое", "рекомендуемое" и т.п. - ?sold # число продаж продукта - inventory # число единиц доступности товара на складе/к заказу; это может быть как отдельное поле, так и сумма доступности всех вариантов продукта - ?verified # признак проверки продукта витриной; витрина может иметь свой "знак качества" и ставить его на продукт, сопровождая это отдельным знаком на карточке или в ленте - variations[] # список объектов вариантов продукта; всегда должен быть минимум один вариант - id # идентификатор варианта на витрине - price # актуальная цена варианта (со скидкой); цены указываются как вещественные числа - msrp # оригинальная цена варианта (без скидки) - ?sold # число продаж варианта; как правило не публикуют - ?inventory # число единиц доступности варианта - ?image # адрес изображения варианта; если при выборе варианта изображение продукта меняется, то это оно - ?color # цвет варианта - ?size # размер варианта **Merchant # данные получаются на странице карточки магазина; все или часть могут дублироваться на карточке продукта** - id # идентификатор магазина на витрине - ?created # дата создания магазина - name # наименование магазина; это поле видит пользователь - ?username # наименование магазина; это поле пользователь не видит, но оно используется для перехода по ссылке в магазин, обычно совпадает с id, но может отличаться - rating # рейтинг магазина - ?ratingCount # число оценок - ?ratingDistr # распределение оценок по баллам - ?itemCount # число продуктов магазина - ?sold # число продаж магазина **Position # данные получаются из лент категорий/магазинов** - productId # идентификатор продукта на витрине; совпадает с Product.id для связи данных Position -> Product - feedType # внутренний код ленты; 1 = лента категории, 2 = лента магазина - feedId # идентификатор категории/магазина на витрине; совпадает с Product.id или Merchant.id - position # номер позиции в ленте **Review # данные получаются на странице карточки продукта, обычно подгружаются тут же или в всплывающем окне** - productId # идентификатор продукта на витрине; совпадает с Product.id для связи данных Position -> Product - id # идентификатор отзыва - created # дата создания отзыва - rating # балл отзыва; обычно это число звёзд от 1 до 5 - ?likesCount # число лайков отзыва - ?moderated # признак модерации отзыва
4	Скорость сбора данных	Данные должны собираться со скоростью, порядка 5 млн. товаров в сутки.
5	Инфраструктура	Разработка парсера ведется на вашей инфраструктуре. Перевод

5	Инфраструктура	Разработка парсера ведется на вашей инфраструктуре. Перевод парсинга в стабильную работу производится уже на нашей инфраструктуре.
6	Формат данных	Дамп данных предоставляется в формате JSON, сжатый gzip архиватором.
7	Язык программирования и используемые компоненты, и модули	На усмотрение исполнителя.
8	Архитектурные требования к системе сбора данных	Архитектура системы сбора данных должна быть выстроена посредством менеджера заданий, раздающего отдельным нодам задания на парсинг. Система сбора данных должна обеспечивать возможность легкого масштабирования за счет подключения нового числа нод. Система собирает разного типа данные (каталог категорий, продукты, магазин, отзывы, позиции в ленте категории/магазина) - модули. Каждый модуль работает независимо друг от друга, и результаты одного модуля (выходные данные) являются заданиями для другого (входные данные). Система работает автономно и непрерывно. Она сама должна создавать для себя новые задания на основе накопленных данных, но при этом должна иметься возможность принимать задания извне. Задания системы должны иметь приоритет. Сбор данных будет происходить заведомо медленнее, чем генерация новых заданий, поэтому будет формироваться приоритетная очередь. Эффективность системы определяется пропускной способностью сбора данных каждым модулем. Система должна иметь возможность простого масштабирования, чтобы по требованию повышать эффективность. Система — это не БД собранных данных. Хранить результат своей работы не является необходимым требованием, собранные данные по готовности сразу же выгружаются потоком во внешнее хранилище, но у системы неизбежно будет служебная БД (на усмотрение исполнителя) на основе части собранных данных для реализации автономной работы и формирования заданий. Как следствие пункта выше, система выгружает данные всегда и только тогда, когда произошло событие сбора данных. каждое задание на входе модуля - это выгрузка свежих данннх на выходе, эффект "кэша" данных недопустим. Система должна уметь эмулировать работу сессии пользователя и работать со списками proxy. Модули лент категории/магазины чувствительны к запросам из разных сессий, в этом случае стабильность выдачи списка продуктов витриной резко снижается. Выгрузка данных осуществляется сообщениями и должна предусматривать различные форматы сериализации, способы сохранения/передачи и их комбинации. Реализация каждого модуля для обеспечения эффективности лежит на усмотрении исполнителя. Система имеет функционал параметров-лимитов сбора данных каждым модулем. Каталог может быть ограничен уровнем вложенности, ленты могут быть ограничены длиной листания. Рабочие процессы (планировщики заданий): Все процессы не конфликтуют друг с другом. Каждый раз, когда

		Все процессы не конфликтуют друг с другом. Каждый раз, когда создаётся задание для модуля любым источником, то при наличии такого же в очереди повторное либо отбрасывается, либо повышает приоритет. Если возникает ситуация, когда за короткий срок создаётся несколько одинаковых заданий, но каждое успевает отработать до постановки нового, то такие повторы допустимы. Главное не дублировать задания, уже имеющиеся в очереди. **1. "полный цикл" (автономная часть):** начинается с модуля каталога категорий - модуль каталога категорий получает список всех категорий с учетом иерархических связей. Каждая найденная категория — это задание для модуля лента категории. - модуль ленты категории получает список продуктов и их позиций в этой ленте. каждый найденный продукт — это задание для двух модулей: продукт и отзывы. - модуль продукта получает информацию о магазине. Каждый найденный магазин — это задание для двух модулей: магазин и лента магазина. - модуль ленты магазина работает аналогично ленте категории. - модуль магазина не генерирует событий. - модуль отзывов не генерирует событий. после завершения всех созданных заданий этим циклом, генерируется повторное на новый полный цикл. **2. "повторный сбор" (автономная часть):** периодически каждый модуль создаёт повторные задания, источником которых является служебная БД. Для каждого модуля может быть создано несколько планировщиков заданий, логика их работы определяется/корректируется в будущем, после того как будет реализован и отлажен "полный цикл". Суть работы таких планировщиков - создать задания на повторный сбор данных по данным, которые не были найдены за время работы нескольких полных циклов. **3. "принудительный сбор" (внешняя часть):** каждый модуль должен быть готов принять извне задания в виде сообщений различными способами. Как правило такие задания имеют более высокий приоритет над автономными планировщиками. прим.: на время разработки формат сообщения выбирается json, а способ передачи - файл(ы), где каждая строка — это сообщение. так же система работает только в полном цикле, и может быть установлен лимит сбора для ускорения тестирования. **прим.: процессы 2. и 3. на первом этапе разработки не реализуются, но их нужно учитывать при проектировании системы.**
9	Сроки разработки системы	1 календарный месяц.