Лабораторная работа №2: Проектирование рекомендательной ленты для социальной сети

Постановка задачи: К вам обратился Product Owner популярной социальной сети, с целью обеспечить доступ пользователя к контенту даже на парковке. Ваша задача – спроектировать архитектуру сервиса ленты постов для социальной сети. Пользователь подписывается на авторов, видит их новые посты в своей персональной ленте, может открывать комментарии и получать уведомления о новых событиях почти в реальном времени.

Функциональные требования:

1. Регистрация/логин, управление профилем.
2. Подписка/отписка на авторов.
3. Публикация постов (текст + опциональные медиа-ссылки).
4. Просмотр персональной ленты с **бесконечной прокруткой** (пагинация по курсору, а не по offset).
5. Добавление и чтение комментариев к постам.
6. Уведомления в реальном времени о новых постах подписок и новых комментариях к моим постам.

Нефункциональные требования:

1. Низкая задержка чтения ленты: p95 < 150 мс для партии из 20 элементов.
2. Высокая пропускная способность записи комментариев (бурсты до 10k RPS без 5xx).
3. Горизонтальное масштабирование ключевых компонентов.
4. Устойчивость к сбоям брокера/воркеров: события не теряются
5. Идемпотентность обработчиков событий и безопасная повторная доставка.

Задания:

1. Оценочные расчёты

- Оцените суточные и пиковые RPS:
    - публикация постов;
    - доставки постов в ленты;
    - чтение ленты (партии по 20);
    - запись/чтение комментариев.
- Оцените объём хранения на 1 и 3 года: посты (метаданные), индексы ленты, комментарии, кэш.
- Заложите бюджет на рост в 3 раза без архитектурных изменений.

2. Модель данных (логическая)

- Реляционный контур (users, posts, subscriptions, outbox_events): ключи, уникальности, транзакционные границы.
- Хранилище ленты
- Комментарии: схема под запрос «N последних по посту, сортировка по времени».
- Опишите, что и где кэшируется для гидратации постов по ID.

3. Технические решения (сравнение и выбор)

- Стратегия формирования ленты
- Пагинация для infinite scroll
- Outbox/Transactional messaging
- Идемпотентность и повторная доставка
- Гидратация: порядок источников (кэш → сервис постов), размеры батчей, таймауты и деградация.
- Бэкап/восстановление и репликация для разных хранилищ.

4. Нарисуйте UML-диаграммы

- **Component / Container** (уровень микросервисов, шина событий, хранилища, кэш).
- **Sequence** (три сценария):
    1. публикация поста с Outbox и доставкой события;
    2. чтение ленты с курсором и гидратацией;
    3. комментарий → событие → WebSocket-уведомление.
- **Deployment** (узлы/контейнеры, масштабирование, внешние зависимости).
- **Class/Data** (логическая модель для реляционной части + стереотипы/примечания для нереляционной).