Схема микросервисов социальной сети с базовыми функциями.

API Gateway

* Центральная точка входа для всех запросов к микросервисам.
* Управляет маршрутизацией запросов, авторизацией и балансировкой нагрузки.

User Service (Сервис пользователей):

* Управляет регистрацией, авторизацией и профилями пользователей.
* Хранит основную информацию о пользователях (имя, email, аватар и т.д.).
* Может содержать управление друзьями/подписками.

Messaging Service (Сервис обмена сообщениями):

* Управляет личными сообщениями между пользователями.
* Включает хранение истории сообщений и функции для управления чатом (создание чатов, удаление сообщений и т.д.).
* Реализует функциональность реального времени для сообщений (например, WebSocket или иной push-механизм).

Post Service (Сервис постов):

* Управляет созданием, редактированием и удалением постов пользователей.
* Содержит информацию о постах, включая контент, время публикации, метаданные (например, теги, геолокацию, ссылки).
* Поддерживает сортировку и фильтрацию постов.

Like Service (Сервис лайков):

* Управляет функционалом лайков для постов.
* Хранит данные о том, кто поставил лайк и на какой пост.
* Отслеживает количество лайков и предоставляет агрегированную статистику.

Notification Service (Сервис уведомлений):

* Отправляет пользователям уведомления о событиях (новое сообщение, лайк на пост, комментарий и т.д.).
* Поддерживает push-уведомления, email и уведомления внутри приложения.

Comment Service (Сервис комментариев):

* Управляет добавлением и отображением комментариев к постам.
* Может включать функциональность для реакции на комментарии и сортировки по популярности.

Analytics Service (Сервис аналитики):

* Сбор данных о действиях пользователей, активности, вовлеченности (например, количество лайков, среднее количество сообщений, посещаемость).
* Служит для генерации отчетов и анализа пользовательской активности.

Auth Service (Сервис аутентификации):

* Реализует вход в систему и управление правами доступа.
* Может использовать JWT или другой метод токенов для обеспечения безопасности.

Event Bus (Шина данных):

* Обеспечивает асинхронное взаимодействие между микросервисами.
* Например, если пользователь поставил лайк на пост, сервис лайков отправит событие, которое затем будет обработано лентой новостей и сервисом уведомлений.

Хранилище данных

* Каждый микросервис может иметь собственное специализированное хранилище данных (SQL/NoSQL), оптимизированное под свои задачи.
* Общие данные могут храниться в распределённом хранилище (например, S3 или Blob).
* Кэширование (Redis): для ускорения доступа к часто запрашиваемым данным, например, лента новостей, лайки и профили пользователей.

Архитектура взаимодействия

* REST и gRPC: для синхронного обмена данными между микросервисами.
* Очереди сообщений (Kafka, RabbitMQ): для асинхронных событий и уведомлений.