**1. План структуры проекта**

* Основной проект Django:
  + settings.py будет настроен для интеграции Redis, Celery, и для работы с сессиями.
* Приложения (apps):
  + **accounts**: управление пользователями, сессиями, аутентификацией.
  + **videos**: загрузка, обработка, хранение и стриминг видео.
  + **comments**: добавление комментариев под видео.
  + **analytics**: сбор и хранение статистики просмотров, лайков и других данных.
  + **notifications**: отправка уведомлений пользователям через Celery (например, о новых комментариях).
  + **search**: реализация поиска видео по тегам и заголовкам.

**2. Приложение accounts**

* **Модели**:
  + User: расширенная модель пользователя.
  + Настройка сессий для хранения данных о входе пользователя.
* **Функционал**:
  + Регистрация/вход.
  + Восстановление пароля.
  + Использование сессий для отслеживания активности пользователя.

**3. Приложение videos**

* **Модели**:
  + Video: модель для хранения метаинформации о видео (название, описание, путь к файлу).
  + VideoProcessingTask: модель для отслеживания задач обработки видео (например, конвертация в нужные форматы через Celery).
* **Функционал загрузки видео**:
  + Видео загружаются через форму на сайт.
  + После загрузки видео передается в Celery для обработки (например, конвертация, создание превью).
  + **Redis** используется для очередей задач.
* **Хранение видео**:
  + Файлы могут храниться в файловой системе или через сторонние сервисы (например, S3).
* **Стриминг видео**:
  + Использовать Django для отдачи статических файлов или настроить nginx для стриминга видеофайлов.

**4. Приложение comments**

* **Модели**:
  + Comment: модель комментариев, привязанных к видео и пользователям.
* **Функционал**:
  + Пользователи могут добавлять комментарии.
  + Комментарии могут модерироваться.

**5. Приложение analytics**

* **Модели**:
  + ViewCount: модель для хранения количества просмотров каждого видео.
  + Like: модель для хранения лайков и дизлайков.
* **Функционал**:
  + Счётчик просмотров увеличивается через Celery при каждом просмотре.
  + Лайки/дизлайки обновляются в реальном времени.

**6. Приложение notifications**

* **Функционал**:
  + Использовать Celery для отправки уведомлений (email или push-уведомления).
  + Уведомления о новых комментариях или ответах.

**7. Приложение search**

* **Функционал**:
  + Реализовать поиск видео по названию, тегам, авторам.
  + Использовать полнотекстовый поиск или сторонние библиотеки для более сложных запросов.

**8. Подключение Celery и Redis**

* **Celery**: для обработки фоновых задач, таких как конвертация видео, отправка уведомлений, обновление аналитики.
* **Redis**: как брокер задач для Celery и кеш для ускорения работы сайта (например, кеширование популярного контента).
* Задания, связанные с видеообработкой, могут включать:
  + Конвертацию видео в несколько форматов (например, для поддержки различных устройств).
  + Создание превью видео.
  + Сжатие больших файлов.

**9. Дополнительно**

* **Аутентификация и авторизация**:
  + Пользователи могут создавать каналы, загружать видео, управлять своими профилями.
* **Пагинация**: для большого количества видео и комментариев.
* **Кеширование**: Redis для ускорения работы сайта, включая кеширование популярных видео и страниц.

**10. Технологии и интеграция:**

* **Видеообработка**: библиотека, например ffmpeg, может использоваться в Celery для конвертации и обработки видео.
* **nginx**: для оптимизации раздачи видеофайлов.