Ниже приведён детальный план из **15 крупных блоков**, включающий в себя **150 пошаговых действий**. Каждый шаг содержит примерную оценку времени (в минутах или часах), чтобы можно было ориентироваться. Учитывается, что у нас есть домохозяйка с минимальными знаниями в программировании, которая может выполнять базовые действия (установка, запуск программ, копирование и вставка кода). Для написания кода и архитектуры мы используем ChatGPT-o1, который даёт не только сам код, но и инструкции по вставке (в какой файл, в какую строку и т.п.).

Мы стремимся получить минимально работающую версию приложения (MVP) примерно через **8 часов**, но общая структура плана рассчитана на дальнейшее развитие проекта с максимальной гибкостью.

**Блок 1. Первоначальная подготовка**

**Шаг 1 (5 минут).**

* Установите или обновите Xcode (последняя версия) из App Store на вашем Mac.
* Оценка времени: 5 минут (плюс время на скачивание, которое может варьироваться).

**Шаг 2 (2 минуты).**

* Создайте аккаунт или войдите в свой Apple ID, который будет использоваться для разработки.
* Оценка времени: 2 минуты.

**Шаг 3 (3 минуты).**

* Убедитесь, что у вас есть базовый план разработчика Apple (бесплатный или платный), чтобы вы могли запускать тестовые сборки на своём iPhone.
* Оценка времени: 3 минуты.

**Шаг 4 (3 минуты).**

* Установите Git (если не установлен). Git обычно идёт вместе с Xcode Command Line Tools, но проверьте, всё ли в порядке.
* Оценка времени: 3 минуты.

**Шаг 5 (5 минут).**

* Создайте папку на компьютере, где будет храниться проект (например, ~/Documents/MyTikTokApp).
* Оценка времени: 5 минут.

**Шаг 6 (3 минуты).**

* Инициализируйте Git-репозиторий командой git init в папке проекта через Терминал.
* Оценка времени: 3 минуты.

**Шаг 7 (5 минут).**

* Создайте на GitHub (или на другом сервисе) новый приватный репозиторий, чтобы хранить код.
* Оценка времени: 5 минут.

**Шаг 8 (5 минут).**

* Свяжите локальный репозиторий с удалённым (git remote add origin ...) и сделайте первый коммит.
* Оценка времени: 5 минут.

**Шаг 9 (5 минут).**

* Откройте Xcode, выберите “Create a new Xcode project” и выберите шаблон “App” (iOS). Назовите проект MyTikTokApp.
* Оценка времени: 5 минут.

**Шаг 10 (2 минуты).**

* Убедитесь, что автоматически сгенерированный проект запускается на симуляторе. Нажмите кнопку «Play» в Xcode.
* Оценка времени: 2 минуты.

**Блок 2. Создание автоматизированного файла архитектуры проекта**

**Шаг 11 (5 минут).**

* Создайте файл ARCHITECTURE.md в корне проекта. В этом файле будем описывать основные модули.
* Оценка времени: 5 минут.

**Шаг 12 (5 минут).**

* Обратитесь к ChatGPT-o1 за шаблоном базовой структуры ARCHITECTURE.md. Пусть он укажет, что в файле будут разделы “Модули”, “Сервисы”, “UI-Компоненты”.
* Оценка времени: 5 минут.

**Шаг 13 (10 минут).**

* Скопируйте результат от ChatGPT-o1 и вставьте в ARCHITECTURE.md. Сделайте коммит в Git.
* Оценка времени: 10 минут.

**Шаг 14 (5 минут).**

* Согласуйте с ChatGPT-o1, каким образом файл ARCHITECTURE.md будет автоматически обновляться. Например, мы можем хранить описание каждого нового файла/класса в виде комментария в коде, а ChatGPT будет генерировать обновлённую архитектуру по запросу.
* Оценка времени: 5 минут.

**Шаг 15 (10 минут).**

* Настройте команду в виде GitHub Action или простой скрипт, который по команде вносит изменения в ARCHITECTURE.md.
* Оценка времени: 10 минут.

**Шаг 16 (10 минут).**

* Обратитесь к ChatGPT-o1 за скриптом (например, на Python или Bash), который будет парсить комментарии в коде и обновлять ARCHITECTURE.md. Пусть ChatGPT-o1 укажет, в какой файл его поместить (например, update\_architecture.py).
* Оценка времени: 10 минут.

**Шаг 17 (5 минут).**

* Скопируйте и вставьте код скрипта в файл update\_architecture.py, согласно указаниям ChatGPT-o1 (например, в строках 1-50).
* Оценка времени: 5 минут.

**Шаг 18 (5 минут).**

* Запустите python update\_architecture.py (или нужную команду) и убедитесь, что ARCHITECTURE.md обновляется.
* Оценка времени: 5 минут.

**Шаг 19 (5 минут).**

* Зафиксируйте (git commit) все изменения.
* Оценка времени: 5 минут.

**Шаг 20 (5 минут).**

* Проверьте, что в ARCHITECTURE.md прописан хотя бы один базовый модуль — например, “Core”, “UI” и “Networking”.
* Оценка времени: 5 минут.

**Блок 3. Планирование MVP (Основной функционал)**

**Шаг 21 (10 минут).**

* Определитесь вместе с ChatGPT-o1, какие основные функции MVP должны появиться за 8 часов:
  1. Регистрация/авторизация
  2. Лента с видеороликами
  3. Возможность загружать короткие видео
  4. Просмотр, лайки (или базовые реакции)
* Оценка времени: 10 минут.

**Шаг 22 (5 минут).**

* Добавьте в ARCHITECTURE.md раздел “MVP Features” и кратко опишите эти функции.
* Оценка времени: 5 минут.

**Шаг 23 (5 минут).**

* Спланируйте вместе с ChatGPT-o1, какие сторонние библиотеки (если нужны) вы будете использовать, например Firebase для простого бэкенда, или встроенные средства iOS для хранения данных локально.
* Оценка времени: 5 минут.

**Шаг 24 (5 минут).**

* Определите структуру данных для хранения коротких видео (например, URL, описание, лайки).
* Оценка времени: 5 минут.

**Шаг 25 (5 минут).**

* Создайте раздел “Data Models” в ARCHITECTURE.md с описанием основных сущностей (пользователь, видео, комментарий и т.д.).
* Оценка времени: 5 минут.

**Шаг 26 (5 минут).**

* Зафиксируйте изменения в Git.
* Оценка времени: 5 минут.

**Шаг 27 (3 минуты).**

* Убедитесь, что общая схема (пользователь, видео, комментарии) описана в ARCHITECTURE.md.
* Оценка времени: 3 минуты.

**Шаг 28 (2 минуты).**

* Переспросите у ChatGPT-o1, всё ли достаточно для старта реализации MVP.
* Оценка времени: 2 минуты.

**Шаг 29 (5 минут).**

* Оцените примерное время выполнения каждой задачи MVP, чтобы уложиться в 8 часов. Запишите это в ARCHITECTURE.md или отдельный план.
* Оценка времени: 5 минут.

**Шаг 30 (2 минуты).**

* Снова зафиксируйте изменения (commit & push).
* Оценка времени: 2 минуты.

**Блок 4. Настройка базовой структуры проекта в Xcode**

**Шаг 31 (5 минут).**

* Создайте новую папку (группу) в Xcode “Core” для моделей и сервисов.
* Оценка времени: 5 минут.

**Шаг 32 (5 минут).**

* Создайте файл UserModel.swift в папке “Core”. Попросите ChatGPT-o1 сгенерировать класс/структуру User (примерно 20 строк). Пусть он укажет:
  + UserModel.swift, строка 1-20.
* Оценка времени: 5 минут.

**Шаг 33 (5 минут).**

* Скопируйте код от ChatGPT-o1 и вставьте в нужное место. Убедитесь, что всё компилируется.
* Оценка времени: 5 минут.

**Шаг 34 (5 минут).**

* Создайте файл VideoModel.swift в папке “Core”. Попросите ChatGPT-o1 сгенерировать структуру для видео.
* Оценка времени: 5 минут.

**Шаг 35 (5 минут).**

* Скопируйте результат (примерно строки 1-30), вставьте в VideoModel.swift.
* Оценка времени: 5 минут.

**Шаг 36 (5 минут).**

* Сделайте коммит с новыми моделями.
* Оценка времени: 5 минут.

**Шаг 37 (5 минут).**

* Создайте папку (группу) в Xcode “Networking”.
* Оценка времени: 5 минут.

**Шаг 38 (10 минут).**

* Попросите ChatGPT-o1 сгенерировать заглушку NetworkingService.swift с методами uploadVideo, fetchVideos. Пусть укажет, где вставлять код (например, в строках 1-50).
* Оценка времени: 10 минут.

**Шаг 39 (5 минут).**

* Скопируйте и вставьте код в NetworkingService.swift.
* Оценка времени: 5 минут.

Пункт оказался очень трудоемким, сюда входила настройка Firebase

**Шаг 40 (5 минут).**

* Запустите проект, убедитесь, что нет ошибок компиляции.
* Оценка времени: 5 минут.

**Блок 5. Подготовка UI: создание экранов и навигации**

**Шаг 41 (5 минут).**

* Создайте в Xcode папку (группу) “UI”.
* Оценка времени: 5 минут.

**Шаг 42 (10 минут).**

* Обратитесь к ChatGPT-o1, чтобы сгенерировать код для MainTabBarController.swift, который будет содержать 2 вкладки — “Home” и “Profile”. Пусть ChatGPT-o1 укажет, куда вставлять.
* Оценка времени: 10 минут.

**Шаг 43 (5 минут).**

* Вставьте сгенерированный код в MainTabBarController.swift (строки 1-60, например).
* Оценка времени: 5 минут.

**Шаг 44 (5 минут).**

* Подключите MainTabBarController в SceneDelegate или AppDelegate, чтобы он был корневым контроллером.
* Оценка времени: 5 минут.

**Шаг 45 (5 минут).**

* Запустите проект. Убедитесь, что видите TabBar с двумя вкладками.
* Оценка времени: 5 минут.

**Шаг 46 (5 минут).**

* Создайте файл HomeViewController.swift и попросите ChatGPT-o1 сгенерировать базовый UI (таблица или коллекция для отображения списка видео).
* Оценка времени: 5 минут.

**Шаг 47 (5 минут).**

* Скопируйте, вставьте код из ChatGPT-o1 в соответствующие строки файла (например, 1-70).
* Оценка времени: 5 минут.
* Тут чуть внесли правки с точки зрения интерфейса, чтобы число лайков и название видео не скрывалось за TabBar

**Шаг 48 (5 минут).**

* Создайте файл ProfileViewController.swift и попросите ChatGPT-o1 сгенерировать простую страницу профиля пользователя (аватар, кнопка выхода и т.д.).
* Тут пришлось почти час налаживать связь с Firebase
* Оценка времени: 5 минут.

**Шаг 49 (5 минут).**

* Скопируйте код из ChatGPT-o1 (строки 1-50) и вставьте в ProfileViewController.swift.
* Оценка времени: 5 минут.

**Шаг 50 (5 минут).**

* Сделайте коммит изменений и проверьте работу во вкладках.
* Оценка времени: 5 минут.

**Блок 6. Реализация загрузки и отображения видео (MVP)**

**Шаг 51 (5 минут).**

* В HomeViewController добавьте кнопку «Обновить ленту» для теста. Попросите ChatGPT-o1 дать код (строки).
* Оценка времени: 5 минут.

**Шаг 52 (5 минут).**

* Подключите кнопку к методу, который вызывает NetworkingService.fetchVideos.
* Оценка времени: 5 минут.

**Шаг 53 (10 минут).**

* Настройте метод fetchVideos так, чтобы он возвращал массив VideoModel. Пусть ChatGPT-o1 сгенерирует код и вставку (строки 30-60, к примеру) в NetworkingService.swift.
* Оценка времени: 10 минут.

**Шаг 54 (5 минут).**

* В методе кнопки вызовите NetworkingService.fetchVideos, сохраните результат в локальном массиве, перезагрузите таблицу или коллекцию.
* Оценка времени: 5 минут.

**Шаг 55 (5 минут).**

* Попросите ChatGPT-o1 уточнить, как связать массив VideoModel с ячейками в таблице/коллекции (если используете UICollectionView, например).
* Оценка времени: 5 минут.

**Шаг 56 (10 минут).**

* Создайте кастомную ячейку (например, VideoCell.swift). Попросите ChatGPT-o1 дать код (вставка в строки 1-70).
* Оценка времени: 10 минут.

**Шаг 57 (5 минут).**

* Зарегистрируйте ячейку в HomeViewController и заполните её данными (название видео, заглушка картинки и т.д.).
* Оценка времени: 5 минут.

**Шаг 58 (5 минут).**

* Запустите и проверьте: при нажатии на «Обновить ленту» появляется список заглушечных видео.
* Оценка времени: 5 минут.

**Шаг 59 (5 минут).**

* Сделайте коммит, зафиксируйте изменения.
* Оценка времени: 5 минут.

**Шаг 60 (5 минут).**

* В ARCHITECTURE.md отметьте, что реализован базовый функционал отображения списка видео.
* Оценка времени: 5 минут.

**Блок 7. Загрузка коротких видео**

**Шаг 61 (5 минут).**

* Создайте кнопку “Add Video” в ProfileViewController или добавьте на TabBar третью вкладку “New Video”.
* Оценка времени: 5 минут.

**Шаг 62 (10 минут).**

* Попросите ChatGPT-o1 сгенерировать контроллер NewVideoViewController.swift, который будет открывать камеру или галерею для выбора видео.
* Оценка времени: 10 минут.

**Шаг 63 (5 минут).**

* Скопируйте код из ChatGPT-o1 (строки 1-80) и вставьте в NewVideoViewController.swift.
* Оценка времени: 5 минут.

**Шаг 64 (10 минут).**

* Настройте Info.plist, добавьте разрешения на доступ к камере и галерее (NSCameraUsageDescription, NSPhotoLibraryUsageDescription).
* Оценка времени: 10 минут.

**Шаг 65 (5 минут).**

* Проверьте работу: при нажатии на кнопку “Add Video” должно появляться окно выбора.
* Оценка времени: 5 минут.

**Шаг 66 (10 минут).**

* Попросите ChatGPT-o1 написать метод uploadVideo(url: URL) в NetworkingService. Вставьте в нужные строки (например, 60-90).
* Оценка времени: 10 минут.

**Шаг 67 (5 минут).**

* В NewVideoViewController при выборе видео вызывайте NetworkingService.uploadVideo(url:) и проверяйте результат (например, в консоль).
* Оценка времени: 5 минут.

**Шаг 68 (5 минут).**

* Сделайте коммит изменений.
* Оценка времени: 5 минут.

**Шаг 69 (5 минут).**

* В ARCHITECTURE.md укажите новый функционал: загрузка видео.
* Оценка времени: 5 минут.

**Шаг 70 (5 минут).**

* Запустите приложение на устройстве (подпишите сертификатом разработчика, если необходимо).
* Оценка времени: 5 минут.

**Блок 8. Авторизация/Регистрация (Firebase или упрощённый вариант)**

**Шаг 71 (5 минут).**

* Решите, использовать ли Firebase (проще для новичка) или свой бэкенд. Предположим, что мы используем Firebase.
* Оценка времени: 5 минут.

**Шаг 72 (10 минут).**

* Создайте проект в [Firebase Console](https://firebase.google.com/). Подключите iOS-приложение (получите GoogleService-Info.plist).
* Оценка времени: 10 минут.

**Шаг 73 (10 минут).**

* Добавьте GoogleService-Info.plist в свой проект Xcode.
* Оценка времени: 10 минут.

**Шаг 74 (5 минут).**

* Установите Firebase SDK через Swift Package Manager или CocoaPods.
* Оценка времени: 5 минут.

**Шаг 75 (5 минут).**

* Инициализируйте Firebase в AppDelegate или SceneDelegate (FirebaseApp.configure()).
* Оценка времени: 5 минут.

**Шаг 76 (10 минут).**

* Создайте файл AuthService.swift и попросите ChatGPT-o1 сгенерировать базовые методы: signUp, signIn, signOut с использованием Firebase Authentication (Email/Password).
* Оценка времени: 10 минут.

**Шаг 77 (5 минут).**

* Скопируйте код из ChatGPT-o1 и вставьте в AuthService.swift (строки 1-70).
* Оценка времени: 5 минут.

**Шаг 78 (10 минут).**

* Создайте простой экран LoginViewController.swift с полями “Email” и “Password” + кнопка “Войти”. Пусть ChatGPT-o1 сгенерирует код.
* Оценка времени: 10 минут.

**Шаг 79 (5 минут).**

* При успешном логине показывайте MainTabBarController.
* Оценка времени: 5 минут.

**Шаг 80 (5 минут).**

* Протестируйте вход/выход. Сделайте коммит.
* Оценка времени: 5 минут.

**Блок 9. Обновление ленты после загрузки нового видео**

**Шаг 81 (5 минут).**

* В uploadVideo (из NetworkingService) добавьте логику записи метаданных видео в Firebase Realtime Database или Firestore.
* Оценка времени: 5 минут.

**Шаг 82 (10 минут).**

* Попросите ChatGPT-o1 скорректировать код в fetchVideos и uploadVideo (строки 30-60, 60-90), чтобы брать данные из Firebase.
* Оценка времени: 10 минут.

**Шаг 83 (5 минут).**

* В HomeViewController замените заглушечные данные на реальные, приходящие из Firebase.
* Оценка времени: 5 минут.

**Шаг 84 (5 минут).**

* Убедитесь, что новое видео сразу появляется в ленте (по крайней мере, после нажатия “Обновить ленту”).
* Оценка времени: 5 минут.

**Шаг 85 (5 минут).**

* Если нужно, добавьте “Pull to Refresh” или автоматическое обновление при открытии вкладки.
* Оценка времени: 5 минут.

**Шаг 86 (5 минут).**

* Сделайте коммит изменений.
* Оценка времени: 5 минут.

**Шаг 87 (5 минут).**

* Добавьте в ARCHITECTURE.md схему взаимодействия UI -> Networking -> Firebase.
* Оценка времени: 5 минут.

**Шаг 88 (5 минут).**

* Запустите скрипт update\_architecture.py, чтобы обновить документ.
* Оценка времени: 5 минут.

**Шаг 89 (5 минут).**

* Проверьте, что MVP функционал (авторизация, просмотр ленты, загрузка видео) готов.
* Оценка времени: 5 минут.

**Шаг 90 (5 минут).**

* Оцените, сколько времени ушло; к этому моменту целевой показатель ~8 часов может быть достигнут.
* Оценка времени: 5 минут.

**Блок 10. Оптимизация UI и базовая анимация**

**Шаг 91 (10 минут).**

* Попросите ChatGPT-o1 улучшить внешний вид HomeViewController с помощью UICollectionViewFlowLayout или UITableView с кастомными ячейками.
* Оценка времени: 10 минут.

**Шаг 92 (5 минут).**

* Вставьте новые значения layout, чтобы ячейки выглядели более-менее как в TikTok (вертикальная прокрутка, полноэкранные ячейки).
* Оценка времени: 5 минут.

**Шаг 93 (5 минут).**

* Настройте автоматический проигрыватель видео при прокрутке (базовая реализация). Обратитесь к ChatGPT-o1.
* Оценка времени: 5 минут.

**Шаг 94 (10 минут).**

* Создайте файл VideoPlayerHelper.swift. Попросите ChatGPT-o1 сгенерировать код, который будет воспроизводить видео из URL.
* Оценка времени: 10 минут.

**Шаг 95 (5 минут).**

* Вставьте код и свяжите с HomeViewController.
* Оценка времени: 5 минут.

**Шаг 96 (5 минут).**

* Сделайте коммит изменений.
* Оценка времени: 5 минут.

**Шаг 97 (5 минут).**

* Запустите на устройстве и протестируйте плавность проигрывания (можно на маленьких роликах).
* Оценка времени: 5 минут.

**Шаг 98 (5 минут).**

* Обратитесь к ChatGPT-o1 за советом, как улучшить анимацию переключения между видео.
* Оценка времени: 5 минут.

**Шаг 99 (5 минут).**

* Добавьте небольшую анимацию при смене ячейки (например, fade-in).
* Оценка времени: 5 минут.

**Шаг 100 (5 минут).**

* Снова зафиксируйте изменения (commit & push).
* Оценка времени: 5 минут.

**Блок 11. Базовые взаимодействия (лайки, комментарии)**

**Шаг 101 (10 минут).**

* Создайте файл InteractionsService.swift в “Networking” для обработки лайков и комментариев.
* Оценка времени: 10 минут.

**Шаг 102 (5 минут).**

* Спросите ChatGPT-o1 сгенерировать методы: likeVideo(videoID: String), commentOnVideo(videoID: String, comment: String).
* Оценка времени: 5 минут.

**Шаг 103 (5 минут).**

* Вставьте код в InteractionsService.swift (строки 1-60).
* Оценка времени: 5 минут.

**Шаг 104 (5 минут).**

* В VideoCell добавьте кнопку “Like”. При нажатии вызывается InteractionsService.likeVideo(...).
* Оценка времени: 5 минут.

**Шаг 105 (5 минут).**

* Запустите приложение, проверьте, что лайк записывается (можно в Firebase).
* Оценка времени: 5 минут.

**Шаг 106 (5 минут).**

* Обновите fetchVideos, чтобы получать актуальное количество лайков.
* Оценка времени: 5 минут.

**Шаг 107 (5 минут).**

* Добавьте базовую кнопку/счётчик лайков на ячейку.
* Оценка времени: 5 минут.

**Шаг 108 (5 минут).**

* Создайте CommentsViewController.swift. Попросите ChatGPT-o1 дать код (строки 1-70).
* Оценка времени: 5 минут.

**Шаг 109 (5 минут).**

* При тапе на кнопку «Комменты» в ячейке открывайте CommentsViewController, загружайте комментарии из Firebase.
* Оценка времени: 5 минут.

**Шаг 110 (5 минут).**

* Зафиксируйте изменения.
* Оценка времени: 5 минут.

**Блок 12. Дополнительные настройки профиля**

**Шаг 111 (5 минут).**

* Добавьте в ProfileViewController поле “Username” и кнопку “Сохранить”.
* Оценка времени: 5 минут.

**Шаг 112 (5 минут).**

* Свяжите это поле с моделью пользователя в Firebase. При смене имени пользовательский профиль обновляется.
* Оценка времени: 5 минут.

**Шаг 113 (5 минут).**

* Спросите ChatGPT-o1, как организовать метод updateUsername в AuthService или UserService.
* Оценка времени: 5 минут.

**Шаг 114 (5 минут).**

* Вставьте полученный код (например, строки 80-90) в AuthService.swift.
* Оценка времени: 5 минут.

**Шаг 115 (5 минут).**

* Добавьте в ProfileViewController отображение загруженных пользователем видео (список).
* Оценка времени: 5 минут.

**Шаг 116 (10 минут).**

* Создайте метод в NetworkingService для получения списка видео конкретного пользователя. Пусть ChatGPT-o1 сгенерирует код.
* Оценка времени: 10 минут.

**Шаг 117 (5 минут).**

* Подключите этот метод и отобразите результат в UITableView/UICollectionView на экране профиля.
* Оценка времени: 5 минут.

**Шаг 118 (5 минут).**

* Сделайте коммит изменений.
* Оценка времени: 5 минут.

**Шаг 119 (5 минут).**

* Запустите скрипт update\_architecture.py, чтобы обновить структуру.
* Оценка времени: 5 минут.

**Шаг 120 (5 минут).**

* Проверите, что ARCHITECTURE.md содержит описание нового функционала.
* Оценка времени: 5 минут.

**Блок 13. Тестирование и отладка MVP**

**Шаг 121 (5 минут).**

* Запустите приложение на реальном устройстве. Протестируйте основные сценарии (авторизация, загрузка видео, просмотр, лайк).
* Оценка времени: 5 минут.

**Шаг 122 (5 минут).**

* Зарегистрируйте ещё одного пользователя и проверьте, что они видят видео друг друга.
* Оценка времени: 5 минут.

**Шаг 123 (5 минут).**

* Зафиксируйте баги или улучшения в виде GitHub Issues (если хотите организовать рабочий процесс).
* Оценка времени: 5 минут.

**Шаг 124 (5 минут).**

* Попробуйте записать короткий 5-секундный ролик с камеры и загрузить его.
* Оценка времени: 5 минут.

**Шаг 125 (5 минут).**

* Проверьте, что ролик воспроизводится без ошибок.
* Оценка времени: 5 минут.

**Шаг 126 (5 минут).**

* Если нужны правки, попросите ChatGPT-o1 помочь с логами и отладкой.
* Оценка времени: 5 минут.

**Шаг 127 (5 минут).**

* По итогам теста внесите небольшие правки (например, UI элементы и т.д.).
* Оценка времени: 5 минут.

**Шаг 128 (5 минут).**

* Сделайте финальный коммит для MVP.
* Оценка времени: 5 минут.

**Шаг 129 (5 минут).**

* Обновите ARCHITECTURE.md, допишите “MVP Completed (v1)”.
* Оценка времени: 5 минут.

**Шаг 130 (2 минуты).**

* Убедитесь, что вы уложились примерно в 8 часов до этого этапа.
* Оценка времени: 2 минуты.

**Блок 14. Дополнительные мелкие улучшения и расширение функционала**

**Шаг 131 (5 минут).**

* Добавьте возможность удаления видео из профиля (добавить кнопку “Delete”).
* Оценка времени: 5 минут.

**Шаг 132 (10 минут).**

* Попросите ChatGPT-o1 дать метод deleteVideo(videoID: String) в NetworkingService (строки 90-120).
* Оценка времени: 10 минут.

**Шаг 133 (5 минут).**

* Вставьте код и проверьте удаление.
* Оценка времени: 5 минут.

**Шаг 134 (10 минут).**

* Создайте элементарную вкладку “Notifications” для уведомлений (можно заглушку). Попросите ChatGPT-o1 помочь.
* Оценка времени: 10 минут.

**Шаг 135 (5 минут).**

* Обновите TabBar на 3 или 4 вкладки (Home, New, Notifications, Profile).
* Оценка времени: 5 минут.

**Шаг 136 (10 минут).**

* Если нужно, настройте Firebase Cloud Messaging. Попросите ChatGPT-o1 сгенерировать базовую интеграцию (уведомления при новых лайках).
* Оценка времени: 10 минут.

**Шаг 137 (5 минут).**

* Вставьте код для FCM (строки, указанные ChatGPT-o1).
* Оценка времени: 5 минут.

**Шаг 138 (5 минут).**

* Сделайте коммит и тестовое уведомление.
* Оценка времени: 5 минут.

**Шаг 139 (5 минут).**

* Обновите ARCHITECTURE.md.
* Оценка времени: 5 минут.

**Шаг 140 (5 минут).**

* Запустите скрипт update\_architecture.py.
* Оценка времени: 5 минут.

**Блок 15. Финальные шаги и развертывание для семьи и друзей**

**Шаг 141 (10 минут).**

* Зарегистрируйте устройства родственников в Apple Developer Center (добавьте их UDID, если нужно).
* Оценка времени: 10 минут.

**Шаг 142 (10 минут).**

* В Xcode настройте Team (ваш Apple ID) и Provisioning Profile для ад-хок распространения или TestFlight (если у вас есть платный аккаунт).
* Оценка времени: 10 минут.

**Шаг 143 (10 минут).**

* Соберите приложение (Archive) и отправьте на TestFlight (или просто ад-хок сборку).
* Оценка времени: 10 минут.

**Шаг 144 (5 минут).**

* Поделитесь ссылкой TestFlight с семьёй и друзьями или отправьте им ад-хок сборку.
* Оценка времени: 5 минут.

**Шаг 145 (5 минут).**

* Убедитесь, что они могут установить и запустить приложение.
* Оценка времени: 5 минут.

**Шаг 146 (5 минут).**

* Попросите их протестировать авторизацию, загрузку и просмотр видео.
* Оценка времени: 5 минут.

**Шаг 147 (5 минут).**

* Соберите от них обратную связь (что работает, что не работает).
* Оценка времени: 5 минут.

**Шаг 148 (5 минут).**

* При необходимости создайте новые задачи для улучшений (GitHub Issues).
* Оценка времени: 5 минут.

**Шаг 149 (5 минут).**

* Снова обратитесь к ChatGPT-o1 с вопросами об оптимизации и масштабировании (если пользователи увеличатся).
* Оценка времени: 5 минут.

**Шаг 150 (5 минут).**

* Завершите создание версии 1.0. Запустите update\_architecture.py в последний раз, чтобы зафиксировать финальное состояние архитектуры.
* Оценка времени: 5 минут.

**Итого**

Данный план содержит **150 шагов** и рассчитан на то, что к моменту около **90-100 шага** (примерно **8 часов** активной работы) у вас уже будет **MVP**: авторизация, лента с видео, загрузка своих роликов и базовые лайки. Оставшиеся шаги расширяют и улучшают функционал.

**Ключевые советы**:

1. **Постоянно обращайтесь к ChatGPT-o1 за кодом**: просите его прямо указывать, куда вставить фрагменты.
2. **Не забывайте делать частые коммиты** и обновлять файл ARCHITECTURE.md (или запускать скрипт update\_architecture.py).
3. **Следите за временем** и разбивайте работу на небольшие участки, как указано в плане.

Таким образом, даже при минимальном опыте программирования вы сможете шаг за шагом создать собственное приложение, близкое по функционалу к TikTok, но ориентированное на узкий круг пользователей (семья и друзья). Удачи в разработке!