План работы с ресурсами API YaMDb

Давайте разберём, **что нужно делать** и **с чего начать**.

Какие ресурсы в API?

Нам нужно реализовать 7 ресурсов:

1. `auth` — аутентификация (регистрация, токены)
2. `users` — пользователи
3. `titles` — произведения
4. `categories` — категории произведений
5. `genres` — жанры произведений
6. `reviews` — отзывы на произведения
7. `comments` — комментарии к отзывам

Важно:

- **`users` и `auth` нужны в первую очередь**, потому что **без аутентификации API не работает**.
- **`titles`, `categories`, `genres` нужны для `reviews`**, так что сначала делаем модели и эндпоинты для них.
- **`reviews` и `comments` зависят от `titles`**, значит, их делаем последними.

Как разделить работу в команде?

Я **(Дима) — аутентификация и пользователи** (`auth`, `users`)
Михаил — произведения, категории, жанры (`titles`, `categories`, `genres`)
Александр — отзывы и комментарии (`reviews`, `comments`)

Какой порядок работы?

Пошаговый план: Сначала делаем модели (`models.py`) для всех ресурсов
Проводим миграции (`makemigrations`, `migrate`)
Реализуем API аутентификации (`auth`)
Реализуем API пользователей (`users`)
Реализуем API произведений, жанров, категорий (`titles`, `genres`, `categories`)
Реализуем API отзывов и комментариев (`reviews`, `comments`)
Тестируем API в Postman и правим баги
Финальный код-ревью и отправка проекта

Что именно делать в API?

1. Реализовать `auth` (аутентификация)

URL:

- `POST /api/v1/auth/signup/` — регистрация (email + username → письмо с кодом)
- `POST /api/v1/auth/token/` — получение токена (username + код → JWT-токен)

Что делать?

- Создать `auth/views.py` и добавить эндпоинты
- Реализовать логику отправки email
- Реализовать генерацию `confirmation_code`
- Реализовать выдачу JWT-токена

---

2. Реализовать `users` (пользователи)

URL:

- `GET /api/v1/users/` — список пользователей (только админы)
- `POST /api/v1/users/` — создание пользователя (админ)
- `GET /api/v1/users/me/` — профиль текущего пользователя
- `PATCH /api/v1/users/me/` — редактирование профиля

- `PATCH /api/v1/users/{username}/` — редактирование пользователя (админ)
- `DELETE /api/v1/users/{username}/` — удаление пользователя

Что делать?

- Создать модель `CustomUser` (`users/models.py`)
- Настроить `permissions.py`, чтобы `user` мог редактировать только себя
- Реализовать вьюсеты и сериализаторы (`users/views.py`, `users/serializers.py`)
- Добавить роуты в `users/urls.py`

---

3. Реализовать `titles`, `categories`, `genres` (Михаил)

URL:

- `GET /api/v1/titles/` — список произведений
- `POST /api/v1/titles/` — создание произведения
- `GET /api/v1/titles/{id}/` — инфо о произведении
- `PATCH /api/v1/titles/{id}/` — редактирование произведения
- `DELETE /api/v1/titles/{id}/` — удаление произведения
- `GET /api/v1/categories/` — список категорий
- `POST /api/v1/categories/` — создание категории
- `DELETE /api/v1/categories/{slug}/` — удаление категории
- `GET /api/v1/genres/` — список жанров
- `POST /api/v1/genres/` — создание жанра
- `DELETE /api/v1/genres/{slug}/` — удаление жанра

Что делать?

- Создать модели `Title`, `Category`, `Genre` (`titles/models.py`)
- Реализовать API в `titles/views.py`
- Добавить роуты в `titles/urls.py`

---

4. Реализовать `reviews` и `comments` (Александр)

URL:

- `GET /api/v1/titles/{title_id}/reviews/` — список отзывов
- `POST /api/v1/titles/{title_id}/reviews/` — добавить отзыв
- `GET /api/v1/titles/{title_id}/reviews/{review_id}/` — получить отзыв
- `PATCH /api/v1/titles/{title_id}/reviews/{review_id}/` — редактировать отзыв
- `DELETE /api/v1/titles/{title_id}/reviews/{review_id}/` — удалить отзыв
- `GET /api/v1/titles/{title_id}/reviews/{review_id}/comments/` — список комментариев
- `POST /api/v1/titles/{title_id}/reviews/{review_id}/comments/` — добавить комментарий
- `GET /api/v1/titles/{title_id}/reviews/{review_id}/comments/{comment_id}/` — получить комментарий
- `PATCH /api/v1/titles/{title_id}/reviews/{review_id}/comments/{comment_id}/` — редактировать комментарий
- `DELETE /api/v1/titles/{title_id}/reviews/{review_id}/comments/{comment_id}/` — удалить комментарий

Что делать?

- Создать модели `Review` и `Comment` (`reviews/models.py`)
- Реализовать API в `reviews/views.py`
- Добавить роуты в `reviews/urls.py`

Заключительный этап

Когда все ресурсы готовы:
Тестируем API в Postman
Делаем код-ревью
Фиксим баги
Обновляем `README.md` и финально пушим в `develop`
Тимлид, тоесть я, мержу `develop` в `master` и отправляю проект на проверку

Итог: что делать прямо сейчас?

Я делаю `auth` и `users`
Михаил делает `titles`, `genres`, `categories`
Александр делает `reviews`, `comments`
Все работают в своих ветках и делают PR в `develop`

Теперь у нас есть чёткий план! За работу!