Задание

Тестовое задание с backend-прокси + хранением своей статистики

## Обновленное задание

Создать приложение с фронтендом и **простым backend-прокси**, который:

- Обязательно работает с реальным API bot.e-replika.ru для получения всех данных (дуа, имена Аллаха, цели, привязки и отметки) — **без моков и локального хранения содержимого контента**.
- При этом backend ведёт собственное **хранение пользовательской статистики и метаданных** (например, время последнего посещения цели, количество повторных открытий дуа/имени, пользовательские заметки к целям).
- Пользовательский токен (test_token_123) используется на backend для запросов к внешнему API.
- Backend реализует агрегирования и аналитику поверх реальных данных API, например:
  - Сколько всего часов пользователь потратил на изучение (подсчёт времени между просмотром позиций).
  - Статистика посещаемости целей, активность по дням.
  - Пользовательские теги/категории для целей.

## Что должен делать backend

- Все CRUD операции с целями, привязками, отметками пересылает на публичный API (прокси).
- Хранит в собственной базе (SQLite) пользовательские данные:
  - last_visited_at для каждой цели/дуа/имени,
  - история просмотров с таймстемпами,
  - пользовательские заметки к целям,
  - любые другие дополнительные метаданные (по желанию).
- Предоставляет отдельные эндпоинты для получения и обновления этих данных для фронтенда.
- Обработка аутентификации с использованием токена, переданного от фронта.

## Что должен делать frontend

- Использовать backend-прокси для получения и модификации основных данных (цели, дуа, имена Аллаха).
- Отдельно запрашивать и отображать статистику/заметки из backend.
- UI для заметок и отображения статистики по изучению.
- Синхронизация состояний между внешним API и локальными метаданными.

## Архитектура и стек

- Backend — любой язык/фреймворк (Node.js/Express, Python/Flask/FastAPI и т.п.), REST API.
- В базе хранить пользовательские данные и метаданные.
- Frontend — SPA (React) с разделением запросов к внешнему API proxy и локальному backend API.

## Технические моменты

- Proxy backend добавляет заголовок Authorization с токеном во все запросы к api `bot.e-replika.ru`.
- При ошибках внешнего API возвращает понятную ошибку с фронтенда.
- Использовать JWT или сессионные куки для авторизации front→backend.

- Настроить CORS для фронтенда.
- Документировать README с инструкциями запуска, настройкой env (токен, URL внешнего API).

## Пример пользовательской статистики для хранения

| ID | user_id | goal_id | dua_id | allah_name_id | last_visited_at | view_count | notes |
|---|---|---|---|---|---|---|---|
| 1 | 1001 | 555 | null | null | 2025-11-29T12:00 | 3 | "Нужно повторить эти имена" |

✅ GET /api/v1/goals/ — список целей
✅ POST /api/v1/goals/ — создание цели
✅ GET /api/v1/goals/{goal_id} — получение цели
✅ PUT /api/v1/goals/{goal_id} — полное обновление
✅ PATCH /api/v1/goals/{goal_id} — частичное обновление
✅ POST /api/v1/goals/{goal_id}/mark — отметка выполненной
✅ POST /api/v1/goals/bind — привязка цели
✅ DELETE /api/v1/goals/{goal_id} — удаление цели
✅ GET /api/v1/goals/bindings — привязки

✅ GET /api/v1/duas/ — список дуа
✅ GET /api/v1/duas/search — поиск дуа
✅ GET /api/v1/duas/{dua_id} — получить дуа по ID

✅ GET /api/v1/allah-names/ — список имен
✅ GET /api/v1/allah-names/random — случайное имя
✅ GET /api/v1/allah-names/number/{number} — имя по номеру
✅ GET /api/v1/allah-names/progress — прогресс изучения
✅ GET /api/v1/allah-names/{content_id} — имя по ID
✅ POST /api/v1/allah-names/{content_id}/view — отметить просмотренным
✅ POST /api/v1/allah-names/{content_id}/mark-learned — отметить изученным
✅ POST /api/v1/allah-names/{content_id}/unmark-learned — снять отметку