Техническое задание

Проект: Веб-аналитика сайта (простая) — аналог Google Analytics (упрощённо)

1. Цель работы

Разработать систему веб-аналитики для сайта: сбор событий посещаемости через небольшой JS-скрипт, сохранение данных на backend в MySQL и отображение метрик в виде дашборда (React).

2. Состав системы

- Tracking-скрипт (JS snippet), который вставляется на отслеживаемый сайт.
- Backend (Node.js) с API для приёма событий и выдачи агрегированных метрик.
- База данных MySQL для хранения сырых событий и агрегатов.
- Dashboard (React) для отображения статистики и графиков.

3. События и данные, которые нужно собирать

- Просмотр страницы (pageview): URL/путь страницы, реферер (источник), время, session_id.
- Сессия: время начала, время последней активности, длительность (в секундах).
- Источник трафика: referrer + UTM-метки (utm_source, utm_medium, utm_campaign).
- Устройство: user-agent (упрощённо: браузер/ОС), тип устройства (desktop/mobile) — опционально.
- География: страна (по IP на сервере).

Сбор персональных данных не требуется. Достаточно session_id (случайный идентификатор) и технических параметров.

4. Требования к tracking-скрипту (JS snippet)

- Подключение одной строкой: <script src=".../tracker.js"></script>
- Автоматически отправлять событие pageview при загрузке страницы.
- Отправлять событие «время на сайте» (например, раз в 10–15 секунд heartbeat) или при закрытии вкладки (sendBeacon).
- Хранить session_id в localStorage или cookie.
- Работать без зависимостей (чистый JS).

5. Backend (Node.js) — приём и хранение

- REST endpoint для приёма событий (например POST /api/track).
- Валидация входных данных и защита от спама (rate limit).
- Определение страны по IP (GeoIP база/библиотека).

- Запись сырых событий в MySQL.
- Периодическая агрегация данных (cron/планировщик) по дням.

6. Метрики, которые должен показывать дашборд

- Посетители в день (уникальные сессии).
- Просмотры страниц (pageviews) в день.
- Среднее время на сайте (по сессиям).
- Процент отказов (bounce rate): сессии с 1 просмотром страницы.
- Источники трафика (рефереры/UTM) — таблица топ-источников.
- Популярные страницы — топ URL по просмотрам.
- География: распределение по странам (таблица + простая карта/визуализация).

7. Интерфейс дашборда (React)

- Главная: карточки KPI (посетители, просмотры, отказ, среднее время).
- График динамики по дням (посетители/просмотры).
- Раздел «Источники» (таблица/диаграмма).
- Раздел «Страницы» (топ страниц).
- Раздел «География» (таблица + карта по странам, можно упрощённую).
- Фильтры: период (7/30/90 дней), (опц.) фильтр по UTM-кампании.

8. Структура БД (MySQL, минимум)

- events: id, session_id, event_type, path/url, referrer, utm_json, user_agent, country, created_at.
- sessions: session_id, first_seen_at, last_seen_at, duration_sec, pageviews_count, country, referrer, utm_json.
- daily_stats: date, visitors, pageviews, bounces, avg_duration_sec.
- daily_pages: date, path, pageviews.
- daily_sources: date, source, visits.
- daily_countries: date, country, visits.

9. Технические требования

- Frontend: React.
- Backend: Node.js (Express/Fastify).
- База: MySQL.
- CORS и безопасность API (минимум: origin whitelist, rate limit).
- README: как подключить tracker.js на сайт, как запустить backend и dashboard.

10. Ограничения

- Достаточно поддержки 1 сайта (multi-site — опционально).
- Cookie banner/согласия не реализуются (для учебного проекта).
- Точность гео допускается приближённая (по стране).

11. Результат работы

- Рабочий трекер, который можно вставить на тестовый сайт и собирать события.
- Backend, который сохраняет данные и отдаёт агрегаты.
- React-дашборд с ключевыми метриками и графиками.
- Демонстрация: посещения тестового сайта отражаются в статистике.

12. Критерии оценки

- Корректный сбор pageview и времени на сайте.
- Стабильное хранение в MySQL и агрегация по дням.
- Понятный дашборд (KPI, графики, таблицы).
- Bounce rate и источники считаются логично.
- Аккуратный код и документация (README).