1) Arquitetura (visão rápida)

2) Estrutura de diretórios

Finalidade: Organização clara do projeto para manutenção.

- web → contém o site e scripts de rastreio.
- db → scripts de inicialização do banco.
- ops → rotinas de backup e automação.

```
project-root/
├─ docker-compose.yml
├─ .env
├─ web/
│  ├─ Dockerfile
│  ├─ vhost/000-default.conf
│  ├─ analytics/
│  │  ├─ config/myphp_config.php
│  │  └─ track.php
│  └─ www/
│     └─ index.html
├─ db/
│  └─ init/001_schema.sql
└─ ops/
   ├─ backup.sh
   └─ crontab
```

3) Arquivo .env

Finalidade: Centralizar senhas, portas e parâmetros (boa prática de DevOps).

```
# WEB
WEB_PORT=8080
PHPMYADMIN_PORT=8081
DB_ROOT_PASSWORD=super_root_secret
DB_NAME=analytics_db
DB_USER=analytics_user
DB_PASSWORD=analytics_pass
HASH_SALT=troque-este-salt-robusto
```

4) docker-compose.yml

Finalidade: Orquestrar os containers (subir tudo com um único comando).

Com esse arquivo, evitamos instalar manualmente PHP, Apache e MariaDB em servidores diferentes. Basta `docker compose up`.

```
version: "3.9"

services:
  web:
    build:
      context: ./web
      dockerfile: Dockerfile
    container_name: analytics_web
    ports:
      - "${WEB_PORT}:80"
    volumes:
      - ./web/www:/var/www/html:ro
      - ./web/analytics:/var/www/html/analytics:ro
      - ./web/vhost/000-default.conf:/etc/apache2/sites-available/000-default.conf:ro
    environment:
      HASH_SALT: ${HASH_SALT}
      PHP_MEMORY_LIMIT: 128M
    depends_on:
      - db
    restart: unless-stopped

  db:
    image: mariadb:10.11
    container_name: analytics_db
    environment:
      MARIADB_ROOT_PASSWORD: ${DB_ROOT_PASSWORD}
      MARIADB_DATABASE: ${DB_NAME}
      MARIADB_USER: ${DB_USER}
      MARIADB_PASSWORD: ${DB_PASSWORD}
    volumes:
      - db_data:/var/lib/mysql
      - ./db/init:/docker-entrypoint-initdb.d:ro
    restart: unless-stopped

  phpmyadmin:
    image: phpmyadmin/phpmyadmin:latest
    container_name: analytics_pma
    environment:
      PMA_HOST: db
      PMA_USER: ${DB_USER}
      PMA_PASSWORD: ${DB_PASSWORD}
    ports:
      - "${PHPMYADMIN_PORT}:80"
    depends_on:
      - db
    restart: unless-stopped
volumes:
  db_data:
```

5) Dockerfile do web

Finalidade: Configurar Apache + PHP de forma endurecida (desabilitar funções perigosas, ativar módulos certos).

Evita brechas de segurança comuns em hospedagens compartilhadas.

```
FROM php:8.2-apache
RUN docker-php-ext-install pdo pdo_mysql
RUN {     echo "expose_php=0";     echo
"disable_functions=exec,passthru,shell_exec,system,proc_open,popen,curl_multi_exec,parse_ini_file,show
_source";     echo "memory_limit=${PHP_MEMORY_LIMIT:-128M}"; } > /usr/local/etc/php/conf.d/custom.ini
RUN a2enmod rewrite headers
RUN chown -R www-data:www-data /var/www/html
USER www-data
```

6) VirtualHost do Apache

Finalidade: Controlar headers de segurança e impedir listagem de diretórios.

```
<VirtualHost *:80>
    ServerAdmin admin@localhost
    DocumentRoot /var/www/html
    <Directory "/var/www/html">
        AllowOverride All
        Require all granted
        Options -Indexes
    </Directory>
    Header always set X-Content-Type-Options "nosniff"
    Header always set X-Frame-Options "SAMEORIGIN"
    Header always set Referrer-Policy "strict-origin-when-cross-origin"
    Header always set Content-Security-Policy "default-src 'self' 'unsafe-inline' 'unsafe-eval' data:
https:"
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

7) Esquema SQL inicial

Finalidade: Estruturar onde as visitas ficam registradas.

```
CREATE TABLE IF NOT EXISTS t_visitas (
  id BIGINT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
  pagina VARCHAR(255) NOT NULL,
  ip_hash CHAR(64) NOT NULL,
  user_agent VARCHAR(512) NOT NULL,
  referrer VARCHAR(512) NULL,
  device VARCHAR(32) NULL,
  navegador VARCHAR(128) NULL,
  pais VARCHAR(64) NULL,
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  KEY idx_pagina (pagina),
  KEY idx_created_at (created_at),
  KEY idx_device (device)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

8) Configuração myphp_config.php

Finalidade: Abstrair a conexão PDO com segurança.
Centralizar a conexão permite mudar credenciais sem mexer no track.php.

```
<?php
$host = "db";
$dbname = getenv("DB_NAME") ?: "analytics_db";
$user = getenv("DB_USER") ?: "analytics_user";
$pass = getenv("DB_PASSWORD") ?: "analytics_pass";
$charset = "utf8mb4";
$dsn = "mysql:host=$host;dbname=$dbname;charset=$charset";
$options = [
  PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
  PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,
  PDO::ATTR_EMULATE_PREPARES => false,
];
try {
  $pdo = new PDO($dsn, $user, $pass, $options);
} catch (PDOException $e) {
  http_response_code(500);
  die("DB connection failed");
}
?>
```

9) Coletor track.php

Finalidade: Capturar e registrar a visita (IP anon, UA, página).
Ele é o coração do sistema — cada visita dispara esse endpoint.

```
<?php
header('Content-Type: image/gif');
header('Cache-Control: no-store, no-cache, must-revalidate, max-age=0');
header('Pragma: no-cache');
function tiny_gif(){ echo
base64_decode('R0lGODlhAQABAPAAAP///wAAACH5BAAAAAAALAAAAAABAAEAAAICRAEAOw=='); }
$ua = $_SERVER['HTTP_USER_AGENT'] ?? '';
$page = $_GET['page'] ?? ($_SERVER['REQUEST_URI'] ?? '/');
$referrer = $_SERVER['HTTP_REFERER'] ?? null;
$ip = $_SERVER['REMOTE_ADDR'] ?? '0.0.0.0';
$device = 'desktop';
if (preg_match('/mobile|android|iphone|ipad/i', $ua)) $device = 'mobile';
$naveg = 'other';
foreach (['Chrome','Firefox','Safari','Edge'] as $b) {
  if (stripos($ua, $b) !== false) { $naveg = $b; break; }
}
$salt = getenv('HASH_SALT') ?: 'change-me';
$ip_hash = hash('sha256', $ip.$salt);
require __DIR__.'/config/myphp_config.php';
try {
  $stmt = $pdo->prepare(
    "INSERT INTO t_visitas (pagina, ip_hash, user_agent, referrer, device, navegador)
     VALUES (?, ?, ?, ?, ?, ?)"
  );
  $stmt->execute([substr($page,0,255), $ip_hash, $ua, $referrer, $device, $naveg]);
} catch (Throwable $e) {}
tiny_gif();
?>
```

10) Snippet no HTML

Finalidade: Inserir no site o código que chama o track.php.

```
<img src="/analytics/track.php?page={ENCODED_PATH}" alt="" width="1" height="1" style="display:none">
<script>
  (function(){
    var p = encodeURIComponent(location.pathname + location.search);
    var el = document.querySelector('img[src^="/analytics/track.php"]');
    if (el) el.src = "/analytics/track.php?page=" + p;
  })();
</script>
```

```
<script>
  (function(){
    var p = encodeURIComponent(location.pathname + location.search);
    fetch("/analytics/track.php?page=" + p, {cache:"no-store",mode:"no-cors"});
  })();
</script>
```