Opção A — com Docker (recomendado: 1 comando e tudo sobe)

1) Estrutura de pastas

```
thermosafe-local/
  docker-compose.yml
  .env
  site/
    index.html (seu site)
    assets/...
    analytics/
      config.php
      track.php
      stats.php
```

2) docker-compose.yml

```
version: "3.9"
services:
  web:
    image: php:8.2-apache
    container_name: thermosafe_web
    ports: ["8080:80"]
    volumes:
      - ./site:/var/www/html
    depends_on: [db]
  db:
    image: mysql:8.0
    container_name: thermosafe_db
    environment:
      MYSQL_ROOT_PASSWORD: ${MYSQL_ROOT_PASSWORD}
      MYSQL_DATABASE: ${MYSQL_DATABASE}
      MYSQL_USER: ${MYSQL_USER}
      MYSQL_PASSWORD: ${MYSQL_PASSWORD}
    ports: ["3306:3306"]
    volumes:
      - dbdata:/var/lib/mysql
  pma:
    image: phpmyadmin:latest
    container_name: thermosafe_pma
    environment:
      PMA_HOST: db
      PMA_USER: ${MYSQL_USER}
      PMA_PASSWORD: ${MYSQL_PASSWORD}
```

```
    ports: ["8081:80"]
    depends_on: [db]
volumes:
  dbdata:
```

Qual é a função de um docker-compose.yml?
É como um **roteiro/receita de cozinha** para o Docke
Em vez de você subir manualmente cada serviço (Apache, PHP, MySQL, phpMyAdmin), o docker-compose.yml descreve **tudo que o projeto precisa** e o Docker Compose orquestra isso de uma vez só.

✅ Apache instalado e configurado
✅ PHP instalado e rodando junto do Apache
✅ Tudo pronto para servir seus .php e .html
Suba um container web com Apache+PHP,
outro container com MySQL,
e outro com phpMyAdmin para gerenciar o banco".

📌 Funções principais do docker-compose.yml

1. Define serviços (containers)
Cada bloco dentro de services: é um container.
web → Apache + PHP (vai rodar seu site e o track.php).
db → MySQL.
pma → phpMyAdmin (interface web para gerenciar o MySQL).
2. Configura portas
"8080:80" → acessa o Apache do container pela porta 8080 do seu PC.
"8081:80" → acessa phpMyAdmin pela porta 8081 do seu PC.
3. Mapeia volumes (arquivos persistentes)
./site:/var/www/html → sua pasta site/ local aparece dentro do container como /var/www/html (document root do Apache).
dbdata:/var/lib/mysql → cria um volume persistente para não perder dados do MySQL quando o container reiniciar.
4. Define dependências
depends_on: [db] → garante que o MySQL suba antes do Apache tentar usar.
5. Gerencia variáveis de ambiente

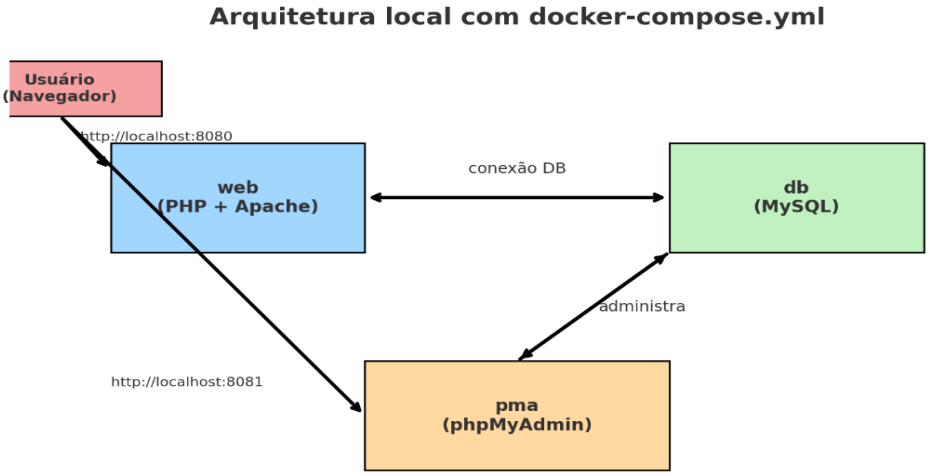
Lê do .env coisas como MYSQL_USER, MYSQL_PASSWORD, MYSQL_DATABASE.

Isso facilita trocar credenciais sem editar o YAML.
6. Orquestra tudo junto
Com **um único comando**:
docker compose up -d
Ele levanta:
Apache + PHP (seu site)
MySQL (banco de dados)
phpMyAdmin (admin do banco)

📌 Benefício para você

Sem instala pacotes, configura serviço, abre porta manualmente.
Se amanhã precisar rodar no servidor da Locaweb (VPS ou dedicado), você já tem a receita pronta.

Facilita para sua equipe: todos rodam o mesmo ambiente, sem diferenças locais.



instalacao de Docker - Para Windows

Passo 1 — Pré-requisitos

Windows 10 Pro/Enterprise/Education **ou** Windows 11 (se for Home, também dá, mas usa WSL2).
Habilitar a virtualização na BIOS (normalmente já vem ativada).

Passo 2 — Instalar WSL2 (se ainda não tiver)

No PowerShell (como admin):
wsl --install
Isso instala o **Subsistema Linux para Windows** e o **kernel**. Reinicie.
Foi definido uma senha: Facil123@. Para o user: Paulo

Passo 3 — Baixar o Docker Desktop

Site oficial: https://www.docker.com/products/docker-desktop/
Clique em **Download for Windows** e instale.

Passo 4 — Testar

Abra o **PowerShell** e rode:
docker --version
docker compose version
Se aparecer a versão, está funcionando.
Versão do Paulo Hor = Docker version 28.3.3, build 980b856

docker compose up -d

PS C:\021-Repositorios\ThermoSafe> docker compose pull
time="2025-09-06T15:13:10-03:00" level=warning msg="C:\\021-Repositorios\\ThermoSafe\\docker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] Pulling 33/33

✓ phpmyadmin Pulled 12.0s
✓ db Pulled 14.5s
PS C:\021-Repositorios\ThermoSafe> docker compose up -d
time="2025-09-06T15:14:48-03:00" level=warning msg="C:\\021-Repositorios\\ThermoSafe\\docker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] Running 4/4

✓ Network thermosafe_default Created 0.0s

✓ Volume "thermosafe_db_data" Created 0.0s
✓ Container phpmyadmin_thermosafe Started
0.4s
✓ Container mysql_thermosafe Started

3) .env

```
MYSQL_ROOT_PASSWORD=rootpass
MYSQL_DATABASE=thermosafe
MYSQL_USER=tsafe
MYSQL_PASSWORD=tsafe123
```

4) site/analytics/config.php

```
<?php
// use variáveis de ambiente em produção se preferir
$db_host = getenv('DB_HOST') ?: 'db';
$db_name = getenv('DB_NAME') ?: 'thermosafe';
$db_user = getenv('DB_USER') ?: 'tsafe';
$db_pass = getenv('DB_PASS') ?: 'tsafe123';

$dsn = "mysql:host={$db_host};dbname={$db_name};charset=utf8mb4";
$options = [
  PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
  PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,
];
$pdo = new PDO($dsn, $db_user, $db_pass, $options);
```

5) site/analytics/track.php

```
<?php
header('Content-Type: application/json');

require __DIR__ . '/config.php';

// cria tabela se não existir
$pdo->exec("
CREATE TABLE IF NOT EXISTS events (
  id BIGINT AUTO_INCREMENT PRIMARY KEY,
```

```
  ts TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  path VARCHAR(255),
  referrer VARCHAR(512),
  ua VARCHAR(512),
  tz VARCHAR(64),
  screen_w INT,
  screen_h INT,
  utm_source VARCHAR(128),
  utm_medium VARCHAR(128),
  utm_campaign VARCHAR(128),
  INDEX idx_ts (ts),
  INDEX idx_path (path)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
");

$raw = file_get_contents('php://input');
$data = json_decode($raw, true) ?: [];

$stmt = $pdo->prepare("
INSERT INTO events (path, referrer, ua, tz, screen_w, screen_h, utm_source,
utm_medium, utm_campaign)
VALUES (:path, :referrer, :ua, :tz, :screen_w, :screen_h, :utm_source, :utm_medium,
:utm_campaign)
");

$stmt->execute([
  ':path' => $data['path'] ?? '/',
  ':referrer' => $data['referrer'] ?? '',
  ':ua' => $_SERVER['HTTP_USER_AGENT'] ?? '',
  ':tz' => $data['tz'] ?? '',
  ':screen_w' => isset($data['screen_w']) ? (int)$data['screen_w'] : null,
  ':screen_h' => isset($data['screen_h']) ? (int)$data['screen_h'] : null,
  ':utm_source' => $data['utm_source'] ?? '',
  ':utm_medium' => $data['utm_medium'] ?? '',
  ':utm_campaign' => $data['utm_campaign'] ?? ''
]);

http_response_code(204);
```

6) (Opcional) site/analytics/stats.php

```
<?php
header('Content-Type: application/json');
require __DIR__ . '/config.php';

$group = $_GET['group'] ?? 'daily';
switch ($group) {
  case 'weekly':  $expr = "DATE_FORMAT(ts, '%x-W%v')"; break; // ISO aprox
  case 'monthly': $expr = "DATE_FORMAT(ts, '%Y-%m')";  break;
```

```
  default:        $expr = "DATE(ts)";
}

$where = [];
$params = [];

if (!empty($_GET['start'])) { $where[] = "ts >= :start"; $params[':start'] = $_GET['start']; }

if (!empty($_GET['end']))   { $where[] = "ts < :end";   $params[':end']   = $_GET['end']; }

if (!empty($_GET['path']))  { $where[] = "path = :path"; $params[':path']  = $_GET['path']; }

$sqlWhere = $where ? ('WHERE ' . implode(' AND ', $where)) : '';

$total = $pdo->prepare("SELECT COUNT(*) AS c FROM events $sqlWhere");
$total->execute($params);
$total = (int)$total->fetch()['c'];

$series = $pdo->prepare("
  SELECT $expr AS bucket, COUNT(*) AS visits
  FROM events
  $sqlWhere
  GROUP BY bucket ORDER BY bucket ASC
");
$series->execute($params);

echo json_encode([
  'total' => $total,
  'group' => $group,
  'series' => $series->fetchAll(),
]);
```

7) Snippet no site (coloque antes de </body>)

```
<script>
(function(){
  try{
    const url = new URL(window.location.href);
    const payload = {
      path: location.pathname,
      referrer: document.referrer || "",
      tz: Intl.DateTimeFormat().resolvedOptions().timeZone || "",
      screen_w: (window.screen && screen.width) || null,
      screen_h: (window.screen && screen.height) || null,
      utm_source: url.searchParams.get("utm_source") || "",
      utm_medium: url.searchParams.get("utm_medium") || "",
      utm_campaign: url.searchParams.get("utm_campaign") || ""
    };
    const send = (body) => {
```

```
    if (navigator.sendBeacon) {
      const blob = new Blob([JSON.stringify(body)], {type:'application/json'});
      return navigator.sendBeacon("/analytics/track.php", blob);
    }
    return fetch("/analytics/track.php", {
      method:"POST", headers:{ "Content-Type":"application/json" },
      body: JSON.stringify(body), keepalive:true
    });
  };
  send(payload);
 }catch(e){}
})();
</script>
```

8) Subir localmente

Instale Docker Desktop (Win/Mac) ou Docker Engine (Linux).

No terminal, dentro de thermosafe-local/:

docker compose up -d

Site: http://localhost:8080

phpMyAdmin: http://localhost:8081 (server: db, user: tsafe, pass: tsafe123)

Migração para Locaweb: só troque as credenciais em config.php (host, db, user, pass) + caminho do snippet (provavelmente continua /analytics/track.php). O resto é igual.

Por que isso te atende hoje

Você testa **tudo local** (site + tracker MySQL) sem depender do painel Locaweb.

Na segunda, você **só sobe os arquivos** e **troca as credenciais**.

Nada de reescrever código ou refatorar — é só **infra/config**.

Quer que eu te entregue esse pacote **zipado** (com docker-compose.yml, .env, config.php, track.php, stats.php, e um README.md curtinho de passos)?

```
docker compose down
docker compose build --no-cache
docker compose up -d
docker os
```

http://localhost:8080
http://localhost:8081