DOCUMENTACION SCRIPT ESTADÍSTICA BÁSICA

GRUPO 1

Contenido

Introducción 3
Requirements 3
Configuración 3
Arranque 4
Colecciones y esquema mínimo 4
Combustible (Combustible) 4
Cabecera (usados): 4
Líneas (dentro de "`líneas`"): 4
Derivados por la API: 4
Vehículo Eléctrico (electrico) 5
Cabecera (usados): 5
Líneas (dentro de lineas): 5
Derivados por API: 5
Peaje (Peaje) 5
Campos usados: 5
Derivados por API: 5
Definición detallada de KPIs 6
Combustible 6
KPIs de usuario 6
KPIs de Empresa 7
Vehiculo eléctrico 8
KPIs de usuario 8
KPIs de Empresa 9
Peaje 10
KPIs de usuario 10
KPIs de Empresa 11
Endpoints 12
Salud y debugs 12
KPIs 13
Query params (opcionales en todos): 13
Helpers de debug recomendados 13
Resolución de problemas (Troubleshooting) 13

Introducción

API en FastAPI que calcula los KPIs necesarios de los tickets almacenado, en nuestro caso en MongoDB Atlas para tres tipos.

- Combustible (“Combustible”)
- Vehiculo eléctrico (“eléctrico”)
- Peaje(“Peaje”)

Incluye también endpoints para debugging, para inspeccionar datos y para configuración.

Requirements

- Python (3.12.10) + → Tambien testado en Python en Python 3.13)
- Mongo DB Atlas o Mongo DB accesible por red.
- Paquetes de Python
    - FastAPI
    - Uvicorn
    - Pymongo
    - Dnspython
    - Pandas
    - Numpy
    - Python-Dotenv

Configuración

Crea un archivo “.env” en el mismo directorio que “main.py”

Bloque de código

```
DB_URL=mongodb+srv://<usuario>:<pass>@<cluster>/<params>
DB_NAME=Prueba1


# (opcionales, si no se definen se usan los defaults)
COL_FUEL=”Combustible”
COL_EV=”eléctrico”
COL_TOLL=”Peaje”
```

La app carga automáticamente este “.env” al iniciar

Arranque

Desde la carpeta donde esta “main.py” ejecutar el script, para luego ejecutar en el terminal el siguiente bloque:

Bloque de código

```
uvicorn main:app –reload
```

- Documentacion interactiva: http://127.0.0.1:8000/docs
- ReDoc: http://127.0.0.1:8000/redoc

Colecciones y esquema mínimo

La API es tolerante: si faltan ciertos campos, genera defaults para que no fallen las agregaciones

Combustible (Combustible)

Cabecera (usados):

- “idTicket”
- “empresaNombre”
- “idUsuario”
- “fechaEmision”
- “horaEmision” *(opcional; default “00:00:00”)*
- “metodoPago”
- “baseImponible”
- “iva”
- “total”
- “lineas” *(lista/dict/string JSON-ish)*

Líneas (dentro de “**líneas**”):

- “producto”
- “litros”
- “precioPorLitro” *(alias: “precio_unitario”, “precio”)*
- “importe” *(opcional; si falta “precioPorLitro”, se usa “importe/litros”)*

Derivados por la API:

- Temporales: “dt”, “mes” formato (YYYY-MM), “hora”, “dia_semana”
- Normalizados líneas: “precio_litro”, “importe_linea”
- Para líneas: “empresaTransporte” = “empresaNombre”

Vehículo Eléctrico (electrico)

Cabecera (usados):

- Igual que combustible

Líneas (dentro de lineas):

- "producto"
- Energía: "kwh" *(alias: "energía"," energia_kwh")*
- Precio: "precio_kwh" *(alias: "precioUnitario", "precio_unitario", "precio")*
- "importe" *(opcional; si falta "precio_kwh", se usa "importe/kwh")*
- "tipoCorriente"/"tarifa", "potenciaKW"/"potenciaMaxKW"/ "power_kw"

Derivados por API:

- Temporales: "dt", "mes", "hora", "dia_semana"
- Normalizados: "kwh", "precio_kwh", "importe_linea"

Peaje (Peaje)

Campos usados:

- "fechaHora"
- "importe" *(puede ser texto: "5,50 €"; la API lo convierte SIEMPRE a float)*
- "idUsuario"
- "empresaNombre"
- "autopista"
- "formaPago"
- "referencia" *(se usa como "idTicket" si no existe)*
- *(opcionales)* "tipoDocumento", "concesionaria", "categoriaVehiculo", "localización", "provincia"

Derivados por API:

- Temporales: "mes", "hora", "dia_semana", "is_weekend"
- "idTicket" (desde referencia si falta)
- "empresa" (copia de "empresaNombre" para agregaciones)

Definición detallada de KPIs

Combustible

KPIs de usuario

- **"gasto_usuario_mes"**
  - Gasto total por usuario mes
  - Agrupación: "sum(total)" por "idUsuario","mes"
  - Salida: [{"idUsuario","mes","total"}]

- **"tickets_usuario_mes"**
  - Numero de tickets por usuario y mes
  - Agrupación → Filtrado desde "lineas": "nunique("idTicket")" por "idUsuario","mes")
  - Salida: [{"idUsuario","mes","tickets"}]

- **"litros_usuario_mes"**
  - Litros por usuario y mes
  - Agrupacion →Filtrado desde "lineas": "nunique("idTicket")" por "idUsuario","mes")

- **"litros_medio_ticket_usuario"**
  - Litros medio por ticket de usuario
  - Calculo
    - Filtrado desde "lineas" → "sum(litros)" por "idUsuario", "idTicket"
    - Media por "idUsuario"
  - Salida: [{"idUsuario","empresaTransporte", "eur_l"}]

- "**precio_usuario_marca**" (Ponderado por litros)
  - Precio en €/L por empresa en base a CPO/estación
  - Agrupacion → Filtrado desde "lineas": media ponderada por "litros" en "idUsuario","empresaTransporte"
  - Salida: [{"idUsuario", "empresaTransporte", "eur_l"}]

- **"métodos_usuario"**
  - Numero de tickets por método de pago para el periodo completo
  - Agrupacion: nunique(idTicket) por "idUsuario", "metodoPago".
  - Salida: [{"idUsuario", "metodoPago", "tickets"}]

- **"metodos_usuario_mes"**
  - Nº de tickets por método y mes, y su % sobre el total mensual del usuario
  - Calculo
    - "tickets = nunique(idTicket)" por "idUsuario", "mes", "metodoPago"

        - "pct = 100 * tickets / sum(tickets por idUsuario, mes)"
    - Salida: [{"idUsuario", "mes", "metodoPago", "tickets", "pct"}]

- **"días_mediana"**
    - Mediana de días entre repostajes
    - Salida: [{"idUsuario", "dias_mediana"}]

- **"anomalias_usuario"**
    - Nº de tickets con líneas cuyo "precio_litro < 0.8" o "> 3.0"
    - Agrupación → Filtrado desde "lineas": "nunique("idTicket)" por "idUsuario"

- **"gasto_dia_semana"**
    - Gasto por dia semana
    - Agrupación → "sum(total") por "idUsuario", "dia_semana".
    - Salida: [{"idUsuario", "dia_semana", "total"}]

KPIs de Empresa

- **"gasto_mes_emp"**:
    - Gasto total por empresa y mes
    - Agrupación: "sum(total)" por "empresaNombre", "mes" → renombrado "empresa".
    - Salida → [{"empresa","mes","total"}]

- **"gasto_total_emp"**:
    - Gasto total en el periodo por empresa
    - Agrupación → sum(total) por "empresaNombre" aplicado a → "empresa"
    - Salida → [{"empresa","gasto_total_periodo"}]

- **"vehiculos_emp"**:
    - Numero de vehículos activos en el periodo por empresa
    - Agrupación → "nunique(idUsuario)" por "empresaNombre" aplicado a "empresa"
    - Salida → [{"empresa","num_vehiculos"}]

- **"gasto_medio_veh":**
    - Gasto medio por vehículo dentro de la empresa
    - Calculo
        - "sum(total)" por "empresaNombre", "idUsuario"
        - media por "empresaNombre"
    - Salida: [{"empresa","gasto_medio_por_vehiculo"}]

- "**ranking_veh" –** (vehiculo): "sum(total)" por "empresaNombre", "idUsuario".

- **“litros_mes_emp”:** Filtrado desde “líneas” → “sum(litros)” por “empresaTransporte”, “mes” aplicado a “empresa”

- **“litros_mes_emp”**: Filtrado desde “líneas” → “sum(litros)” por “empresaTransporte”, “mes” aplicado a “empresa”

- “**precio_global_emp**” (ponderado): Filtrado desde “lineas” → media ponderada por “litros” en “empresaTransporte”.

Vehiculo eléctrico

KPIs de usuario

Básicamente igual que combustible, únicamente cambiando “litros” por → “kWh” y “eur_l” por → eur_kwh”

- “**gasto_usuario_mes”:** “sum(total)” por “idUsuario”, “mes”.

- **“tickets_usuario_mes”**: “nunique(idTicket”) por “idUsuario”, “mes”.

- **“kwh_usuario_mes”**: Filtrado por “líneas” → “sum(kwh”) por “idUsuario”, “mes”

- **“kwh_medio_ticket_usuario**” : Filtrado por “lineas” → “sum(kwh)” por “idUsuario”, “idTicket” → media por “idUsuario”.

- “**precio_usuario_cpo**” (ponderado): Filtrado por “líneas” → media ponderada por “kwh” en “idUsuario”, “empresaTransporte”.

- **“metodos_usuario”**, “**metodos_usuario_mes”**, “**dias_mediana”**, “**gasto_dia_semana”**

    - Básicamente igual a combustibles , mismo calculo aplicado a coche eléctrico

- **“anomalias_usuario”**: Filtrado por “líneas” con “precio_kwh < 0.20” o “> 1.50” → “nunique(idTicket”) por “idUsuario”.

KPIs de Empresa

- **"gasto_mes_emp"**
  - Gasto total por empresa y mes.
  - Agrupación: "sum(total)" por "empresaNombre", "mes" → renombrar "empresaNombre" a "empresa".
  - Salida: [{"empresa","mes","total"}]

- **"gasto_total_emp"**
  - Gasto total del período por empresa.
  - Agrupación: "sum(total)" por "empresaNombre" → "empresa".
  - Salida: [{"empresa","gasto_total_periodo"}]

- **"vehiculos_emp"**
  - **N**úmero de vehículos (usuarios) activos por empresa en el período.
  - Agrupación: "nunique(idUsuario)" por "empresaNombre" → "empresa".
  - Salida: [{"empresa","num_vehiculos"}]

- **"gasto_medio_veh"**
  - Gasto medio por vehículo dentro de cada empresa.
  - **S**umar "total" por "empresaNombre,idUsuario" y luego hacer media por "empresaNombre (cabecera)" → "empresa".
  - Salida: [{"empresa","gasto_medio_por_vehiculo"}]

- **"ranking_veh"**
  - Gasto por vehículo para ranking interno de la empresa.
  - Agrupación: "sum(total" por "empresaNombre","idUsuario" → "empresa".
  - Salida: [{"empresa","idUsuario","gasto_usuario"}]

- **"kwh_mes_emp"**
  - Energía total (kWh) por empresa y mes.
  - Agrupación: "sum(kwh)" por "empresaTransporte,mes" Filtrado desde "líneas"→ renombrar "empresaTransporte" a "empresa".
  - Salida: [{"empresa","mes","kwh"}]

- **"kwh_veh_emp"**
  - "kWh" consumidos por vehículo dentro de cada empresa.
  - Agrupación: "sum(kwh") por "empresaTransporte,idUsuario" Filtrado desde "líneas" → "empresa".
  - Salida: [{"empresa","idUsuario","kwh"}]

- **“precio_global_emp” (ponderado por kWh)”**

  - Precio efectivo medio en €/kWh de la empresa (ponderado por energía).
  - Agrupación**;** Media ponderada por “kwh” en “empresaTransporte” Filtrado desde “líneas”: → “empresa”.
  - Salida**:** `[{"empresa","eur_kwh"}]`

Peaje

La API convierte siempre “importe” a “float” (acepta “5,50 €”, “1.234,56 €”, etc.).

KPIs de usuario

- **“gasto_usuario_mes”**

  - Gasto de peajes por usuario y mes
  - Agrupación: `sum(importe)` por `idUsuario, mes`
  - Salida: `[{"idUsuario","mes","gasto_mes"}]`

- **“tickets_usuario_mes”**

  - Nº de pasos/tickets por usuario y mes
  - Agrupación: “nunique(idTicket)” por “idUsuario”, “mes” (si falta, “idTicket” usa “referencia”)
  - Salida:`[{"idUsuario","mes","tickets"}]`

- **“usuario_autopista_mes”**

  - Desglose por autopista dentro de cada mes del usuario
  - Cálculo (por “idUsuario”, “mes”, “autopista”)
    - “gasto_autopista_mes” = sum(importe)
    - “tickets_autopista_mes” = count()
    - “coste_medio_ticket” = “gasto_autopista_mes” / “tickets_autopista_mes”
    - “pct_gasto_usuario_mes” = “100 * gasto_autopista_mes” / “sum(gasto_autopista_mes” del “usuario” en el mes)
  - Salida: `[{"idUsuario","mes","autopista","gasto_autopista_mes","tickets_autopista_mes","coste_medio_ticket","pct_gasto_usuario_mes"}]`

- **“metodos_usuario_mes”**

  - Nº de tickets por método y mes + % mensual
  - Cálculo
    - “tickets” = “nunique(idTicket”) por “idUsuario”, “mes”, “formaPago”
    - “pct” = 100 * “tickets” / “sum(“tickets” por “idUsuario”, “me”s)
  - Salida: `[{"idUsuario","mes","formaPago","tickets","pct"}]`

- **“dias_mediana”**

  - Mediana de días entre pasos de peaje
  - Agrupación/Calc: mediana de diferencias de “fechaHora” por “idUsuario”
  - Salida: [{"idUsuario","dias_mediana"}]

- **“pct_finde_usuario”**

  - % de pasos por peaje que caen en fin de semana
  - Agrupación: mean(“is_weekend”)*100 por “idUsuario”
  - Salida: [{"idUsuario","pct_finde"}]

KPIs de Empresa

- **“gasto_mes_emp”**

  - Gasto total por empresa y mes
  - Agrupación: “sum(importe)” por “empresa”, “mes”
  - Salida: [{"empresa","mes","gasto_mes"}]

- **“tickets_mes_emp”**

  - Nº de tickets por empresa y mes
  - Agrupación: “nunique(“idTicket”)” por “empresa”, “mes”
  - Salida: [{"empresa","mes","tickets_mes"}]

- **“emp_autopista_mes”**

  - Desglose por autopista dentro de cada mes de la empresa
  - Cálculo (por “empresa”, “mes”, “autopista”)
    - “gasto_autopista_mes” = “sum(importe)”
    - “tickets_autopista_mes” = “count()”
    - “coste_medio_ticket” = “gasto_autopista_mes” / “tickets_autopista_mes”
    - “pct_gasto_empresa_mes” = “100 * gasto_autopista_mes” / “sum(“gasto_autopista_mes” de la “empresa” en el “mes”)
  - Salida: [{"empresa","mes","autopista","gasto_autopista_mes","tickets_autopista_mes","coste_medio_ticket","pct_gasto_empresa_mes"}]

- **“met_emp_mes”**

  - Nº de tickets por método y mes + % mensual (“empresa”)
  - Cálculo
    - “tickets” = “nunique(“idTicket”) por “empresa”, “mes”, “formaPago”
    - “pct = 100 * “tickets” / “sum(“tickets” por “empresa”, “mes”)
  - Salida: [{"empresa","mes","formaPago","tickets","pct"}]

- **“gasto_total_emp”**
    - Gasto total en el periodo por empresa
    - Agrupación: “`sum(importe”)` por “`empresa”`
    - Salida : `[{"empresa","gasto_total_periodo"}]`
- **“vehiculos_emp”**
    - Nº de vehículos (usuarios) activos por empresa
    - Agrupación: “`nunique(“idUsuario”)` ” por “`empresa”`
    - Salida: `[{"empresa","num_vehiculos"}]`
- **“gasto_medio_veh”**
    - Gasto medio por vehículo dentro de la empresa
    - Cálculo: “`sum(“importe”)` ” por “`empresa`,” “`idUsuario”` y media por “`empresa”`
    - Salida:`[{"empresa","gasto_medio_por_vehiculo"}]`
- **“pct_finde_emp”**
    - % de pasos en fin de semana por empresa
    - Agrupación: “`mean(“is_weekend”)*100` por “`empresa”`
    - Salida: `[{"empresa","pct_finde"}]`

Endpoints

Salud y debugs

- “`GET /health”` — resumen y conteos de colecciones
- “`GET /debug/config”` — DB/colecciones/config
- “`GET /debug/peek?n=3”` — columnas y muestra por colección
- “`GET /debug/distinct?collection=<Colección>&field=<campo>&limit=50”` — distintos y top conteos
- “`GET /debug/peaje_sample?n=5”` — sample crudo de Peaje
- “`GET /debug/peaje_after?...”` — cómo queda Peaje tras normalizar (conversión de importe, fechas, etc.)

KPIs

- "GET /kpis/combustible"
- "GET /kpis/ev"
- "GET /kpis/peaje"

Query params (opcionales en todos):

- "start_date" — Formato "YYYY-MM-DD"
- "end_date" — Formato "YYYY-MM-DD"
- "empresa" — filtra por "empresaNombre"
- idUsuario — filtra por "idUsuario"

Helpers de debug recomendados

- **Estado rápido**:
  GET /health
- **Config y conteos**:
  GET /debug/config
- **Muestra por colección**:
  GET /debug/peek?n=5
- **Distribución de valores**:
  GET /debug/distinct?collection=Peaje&field=idUsuario
- **Peaje normalizado**:
  GET /debug/peaje_after?n=5

Resolución de problemas (Troubleshooting)

• Could **not import module "main"**: instala dependencias ("python-dotenv", "pandas", "numpy", …) y ejecuta uvicorn desde el venv correcto.

• Internal **Server Error en Peaje**: suele ser "importe" en string. La API ya lo convierte SIEMPRE; si persiste, revisa "/debug/peaje_after" y asegúrate de que "importe_all_nan" es "false".

• Colecciones vacías: confirma nombres exactos en ".env" ("Combustible", "eléctrico", "Peaje") y la "DB_NAME".

• Fechas: formato "YYYY-MM-DD". Para Peaje, "end_date" incluye hasta "23:59:59" del día.