# ANEXO · TAREAS DEL ALUMNADO

## DATOS, FILTROS Y TOTALES: UNA RUTA DE INFORME EN SVELTEKIT

Nombre: _____ Fecha: _____

Hoy rellenas las bandas del martes con datos reales, en tu app. Piensa en una cocina: los ingredientes vienen de distintas tiendas (**fuentes**: JSON, CSV, base de datos, API); los lavas (**convertir**), eliges (**filtrar**), cocinas (**calcular**) y emplatas (**agrupar y totalizar**).

## PASO 1 · PREPARAR EL PROYECTO Y CARGAR EL JSON (7 MIN)

Sigue al profesor en tu propio proyecto. Descarga los dos ficheros de Classroom y colócalos así:

```
static/data/notes.csv            ← 22012027_notes.csv, renombrado
src/lib/data/folders.json        ← 22012027_folders.json, renombrado
src/routes/report/+page.svelte   ← nuevo, vacío
npm install d3-dsv
```

| OK | HAZ | ESTÁ HECHO CUANDO... |
|---|---|---|
| ☐ | Importa `folders.json` en `+page.svelte` y muéstralo en una lista con `{#each}`. | `/report` muestra las cuatro carpetas. |

## PASO 2 · CSV Y UN VALOR CALCULADO (24 MIN)

| OK | HAZ | ESTÁ HECHO CUANDO... |
|---|---|---|
| ☐ | Crea `src/routes/report/+page.js` con una función `load` que hace `fetch('/data/notes.csv')` y lo pasa por `csvParse`. | La página dice **36 notes**. |
| ☐ | Crea `src/lib/report.js` con `toNote(row)`, que convierte cada fila a tipos reales, y pásala como segundo argumento a `csvParse`. | La suma de palabras da **9213**, no un texto larguísimo. |
| ☐ | Añade el valor calculado `minutes` (palabras ÷ 238, redondeado hacia arriba con `Math.ceil`). | WCAG contrast examples (526 palabras) da **3** minutos. |
| ☐ | Pinta la tabla con cuatro celdas: título, creada, palabras, minutos. | Ves las 36 filas. |

## PASO 3 · FILTROS Y PARÁMETROS (15 MIN)

Un **filtro** decide qué filas entran en el informe. Un **parámetro** es el valor que el usuario da al filtro.

| OK | HAZ | ESTÁ HECHO CUANDO... |
|---|---|---|
| ☐ | Escribe `filterNotes` en `report.js`. Los parámetros son `$state`; las filas, `$derived`. | La tabla recorre `rows`, no `data.notes`. |
| ☐ | Añade los tres controles: carpeta, creada desde y solo fijadas. | Al cambiar un control, la tabla cambia sola. |
| ☐ | Añade un botón **Reset filters** con una función `reset()`: `folder = 'All'`, `from = ''`, `pinnedOnly = false`. | Vuelves a ver las 36 notas. |
| ☐ | Prueba **Work** + **Pinned only** + desde **01/01/2027** y añade el estado vacío (estado **C · Empty** del Figma). | Sale «No notes match these filters.» en lugar de una tabla vacía. |

## PASO 4 · RECUENTOS, TOTALES Y GRUPOS (18 MIN)

| OK | HAZ | ESTÁ HECHO CUANDO… |
|---|---|---|
| ☐ | Escribe `totals(rows)`: recuento, suma de palabras, suma de minutos y media. | Con cero filas, la media es 0, no `NaN`. |
| ☐ | Añade una fila `<tfoot>`: el **report footer**. | Los totales cambian con el filtro. |
| ☐ | Escribe `groupByFolder(rows)` con `Object.groupBy` y la **misma** función `totals`. Un `<tbody>` por grupo. | Cada carpeta tiene su group header y su group footer con subtotales. |

## PASO 5 · COMPRUEBA TUS NÚMEROS (10 MIN)

Prueba las seis combinaciones y compara con tu pantalla:

| OK | FILTRO | NOTAS | PALABRAS | MIN |
|---|---|---|---|---|
| ☐ | Sin filtro | 36 | 9213 | 58 |
| ☐ | Folder = School | 12 | 5217 | 28 |
| ☐ | Pinned only | 9 | 2451 | 15 |
| ☐ | Created from 01/12/2026 | 10 | 2952 | 17 |
| ☐ | School + from 01/12/2026 | 4 | 1833 | 9 |
| ☐ | Work + pinned + from 01/01/2027 | 0 | estado vacío | |

Si algo falla:

- **Las palabras salen como 0419485…** No pasas `toNote` a `csvParse`, o te falta `Number()`.
- **Todas las notas salen fijadas.** Usa `row.pinned === 'true'`, no `row.pinned`: el texto `"false"` no está vacío, así que cuenta como verdadero.
- **Los totales no cambian con el filtro.** Escribiste `totals(data.notes)` en lugar de `totals(rows)`.
- **El total de minutos da 39, no 58.** Dividiste el total de palabras en lugar de sumar los minutos redondeados de cada fila. El footer debe cuadrar con la columna que tiene encima.
- **Error 404 con el CSV.** El fichero está en `src/` en lugar de `static/data/`.

**Tu propio informe (4 min).** Abre tu wireframe del martes. Crea el fichero de datos de tu informe (de 10 a 30 filas, JSON o CSV) y anota qué funciones de `report.js` copiarás y renombrarás.

Fuente de mi informe: _____

Línea de mi ficha que es el filtro: _____

## PREGUNTAS DE LA SESIÓN

**1** La misma nota está en CSV y en JSON. ¿En cuál de los dos 526 es un número? ¿Por qué?

**2** ¿Por qué no basta con `line.split(',')` para leer un CSV?

**3** Las medias de palabras de las cuatro carpetas son 182, 435, 41 y 259. ¿Cuál es la media de todo el informe y cómo se calcula bien?

## ENTREGA

En la tarea de Classroom **UT6 · Report with filters**: (1) una captura del navegador con el estado **B** (School, desde 01/12/2026: 4 notas · 1833 palabras · 9 min); (2) tu `report.js`; (3) una línea de texto: ¿cuál de tus valores es calculado, y a partir de qué columnas?

## PARA CASA

- Termina tu propio informe hasta este punto: tu fichero de datos, filas convertidas, un valor calculado, un filtro con su estado vacío, subtotales por grupo y totales generales. Debe coincidir con tu wireframe del 19/01; si el código te obligó a cambiar el diseño, actualiza el wireframe y explica por qué.
- Ejercicio de transferencia (un párrafo): una app de gimnasio tiene tres clases: yoga (40 personas, edad media 50), spinning (10 personas, edad media 20) y boxeo (10 personas, edad media 20). El report footer dice «edad media: 30». ¿Es correcto? ¿Qué debería decir, y qué hizo mal el programador?