```
import express from "express";
import fs from "fs";
import csv from "csv-parser";

const app = express();
const PORT = 3000;

let parts = [];

// --- 1️⃣ Lae andmed CSV failist mällu ---
function loadData() {
  console.log("Laen andmeid failist LE_5MB_new_format.txt ...");
  return new Promise((resolve, reject) => {
    const results = [];
    fs.createReadStream("LE_5MB_new_format.txt")
      .pipe(csv())
      .on("data", (row) => {
        // teisendame numbrid
        const warehouses = [
          Number(row.werehous1),
          Number(row.werehous2),
          Number(row.werehous3),
          Number(row.werehous4),
          Number(row.werehous5),
        ];
        const totalQty = warehouses.reduce((a, b) => a + b, 0);

        results.push({
```

```
        Serial: row.Serial,
        Name: row.Name,
        Warehouses: warehouses,
        MinPrice: parseFloat(row["min.price"]),
        MaxPrice: parseFloat(row["max.price"]),
        Qty: totalQty,
        Category: row.Category,
      });
    })
    .on("end", () => {
      parts = results;
      console.log(`✅ Laetud ${parts.length} kirjet mällu.`);
      resolve();
    })
    .on("error", reject);
  });
}

// --- 2️⃣ API: Andmete väljastamine ---
app.get("/parts", (req, res) => {
  let { search, page = 1, limit = 50, sort } = req.query;
  page = Number(page);
  limit = Number(limit);

  let filtered = parts;

  // 🔍 Otsi nime või seerianumbri järgi
  if (search) {
```

```
  const s = search.toLowerCase();
  filtered = filtered.filter(
    (p) =>
      p.Name.toLowerCase().includes(s) ||
      p.Serial.toLowerCase().includes(s) ||
      p.Category.toLowerCase().includes(s)
  );
}

// ↕ Sorteerimine (nt ?sort=price_asc või ?sort=name_desc)
if (sort) {
  const [field, order] = sort.split("_");
  filtered.sort((a, b) => {
    let valA, valB;
    switch (field) {
      case "price":
        valA = a.MinPrice;
        valB = b.MinPrice;
        break;
      case "name":
        valA = a.Name;
        valB = b.Name;
        break;
      case "qty":
        valA = a.Qty;
        valB = b.Qty;
        break;
      default:
```

```
     return 0;
   }
   if (order === "desc") return valB > valA ? 1 : -1;
   else return valA > valB ? 1 : -1;
  });
 }

 // 📄 Lehekülgedeks jagamine
 const start = (page - 1) * limit;
 const end = start + limit;
 const paged = filtered.slice(start, end);

 res.json({
  total: filtered.length,
  page,
  limit,
  results: paged,
 });
});

// --- 3️⃣ Käivita server ---
loadData().then(() => {
 app.listen(PORT, () => {
  console.log(`🚀 Server töötab: http://localhost:${PORT}/parts`);
 });
});
```