FindScan – Frontend Intern Assignment: Bollinger Bands (KLineCharts)

Goal: Build a production-ready Bollinger Bands indicator for our charting module using **KLineCharts** only. The indicator must expose the specified settings (with defaults), render correctly, and feel close to TradingView in behavior and UI.

---

1) Tech Constraints

- **Required:** React + Next.js + TypeScript + TailwindCSS + **KLineCharts** (indicator).
- **Do not** use any other charting library or paid component.
- You may use any basic helper/math packages if needed (or write your own small utilities).

---

2) Features & Settings (all mandatory)

Implement **Bollinger Bands** with the following user-configurable settings. **All settings are mandatory** and must have the defaults below.

Inputs

- **Length**: `20`
- **Basic MA Type**: `SMA` *(for this assignment, SMA support is sufficient; expose the field with **`SMA`** as the default)*
- **Source**: `Close` *(use close price)*
- **StdDev (multiplier)**: `2`
- **Offset**: `0` *(shift the bands by N bars; positive values shift forward)*

Style

Provide a **Style** tab similar to TradingView with at least:

- **Basic (middle band)**: visibility toggle + color + line width + line style (solid/dashed)
- **Upper band**: visibility toggle + color + line width + line style
- **Lower band**: visibility toggle + color + line width + line style

- **Background fill** (area between Upper & Lower): visibility toggle + opacity

  UI reference: In TradingView → Indicators → Bollinger Bands → *Settings* (Inputs/Style). Replicate the spirit and simplicity of that UI using Tailwind (no need to pixel-match).

---

3) Data

- Use **demo OHLCV data** (CSV/JSON) with at least **200 candles** on any reasonable timeframe (e.g., 1m/5m/1D).
- Render a standard candlestick series plus the Bollinger Bands overlay.

---

4) Calculations (Expected Formulas)

- **Basis (middle band)** = `SMA(source, length)`
- **StdDev** = standard deviation of the last `length` values of `source` (document clearly whether you used population or sample; either is acceptable if consistent)
- **Upper** = `Basis + (StdDev multiplier * StdDev)`
- **Lower** = `Basis - (StdDev multiplier * StdDev)`
- **Offset**: shift the three series by `offset` bars on the chart

Recompute bands on **every data update** and **on every input change**. Updates should feel instantaneous on the dataset size above.

---

5) UX Expectations

- Indicator can be **added once** via a simple button/menu.
- A **Settings** modal/panel with two tabs: **Inputs** and **Style**.
- Changing any setting updates the chart immediately (no page refresh).
- Sensible default colors; respect dark backgrounds.
- Tooltip/crosshair should show Basis/Upper/Lower values for the hovered candle.

---

6) Deliverables

1. **GitHub repository** (public or view-access link) or a **ZIP** with:
    - Source code (Next.js + TS).
    - `README.md` including:
        - Setup/run instructions (`npm i && npm run dev`).
        - Short note on formulas and which StdDev variant you used.
        - KLineCharts version.
        - Two screenshots or a short GIF of the indicator + settings.
2. (Optional) **Hosted demo** (Vercel/Netlify) – helpful but not required.

---

7) Acceptance Criteria

- **Correctness**: Bands match expected behavior for the given inputs; Basis tracks MA, Upper/Lower expand/contract with volatility; Offset shifts bands correctly.
- **UI/UX**: Clean, simple settings UI inspired by TradingView; visibility/opacity/line controls function as expected.
- **Performance**: Smooth interaction on 200–1,000 candles (no jank on settings changes).
- **Code Quality**: Type-safe, modular (e.g., a small `computeBollingerBands()` utility), readable structure, minimal coupling to page components.
- **KLineCharts Only**: No alternative charting libraries.

---

8) Suggested Project Structure (example)

```
/ (Next.js app)
 /app
  /page.tsx              # renders chart + add-indicator button + settings modal
 /components
  Chart.tsx              # wraps KLineCharts init & updates
  BollingerSettings.tsx    # inputs + style UI
 /lib
  indicators/
   bollinger.ts          # computeBollingerBands(data, options)
  types.ts               # OHLCV & indicator types
 /public/data/ohlcv.json    # demo data
 README.md
```

---

9) Submission

- Share the **repo/ZIP** and (if available) **live demo link**.

- Include a brief note on any trade-offs or known issues.
- **Deadline:** Submit within **3 days** of receiving this assignment.

---

10) Reference Links

- **KLineCharts official docs/demo:** https://klinecharts.com/en-US/
- **TradingView (for UI reference):** https://www.tradingview.com/chart

---

11) Notes

- Feel free to structure components your way; the outline above is just an example.
- Keep the scope to the features above. If you'd like to add more polish, keep it isolated and documented.