The error you're seeing during `npm install` (or `yarn install`) comes from an **extremely outdated dependency chain** in your project.

Here's the root cause breakdown:

- Your project (likely a React app called "letters-social") depends — directly or indirectly — on `google-gax`, which in turn depends on an ancient version of `grpc` → `grpc@1.13.1`.
- `grpc@1.13.1` was released around 2018 and only supports Node.js versions up to ~10–12 at best. It is **completely incompatible** with modern Node.js (especially v20.19.0).
- Two main failure modes appear in your log:
  1. **Pre-built binary download fails** with 403 (Google no longer hosts binaries for such old versions + Node 20 ABI v115 doesn't exist in their old storage).
  2. **Source compilation fails** via node-gyp because:
     - The C++ code doesn't compile cleanly with modern clang (Apple's compiler on macOS Sonoma/Ventura/etc.).
     - It triggers `-Werror` (warnings treated as errors), specifically the string literal concatenation warning in `stats_data.cc:284`.
     - Older `make` steps also fail (missing .deps files, etc.).

grpc@1.x **has been deprecated for many years**. The modern replacement is `@grpc/grpc-js` (pure JavaScript, no native compilation needed).

Recommended Fixes (choose one, ordered from easiest to most thorough)

1. **Best quick fix if you don't actually need Google Cloud libraries right now**
   Remove or replace the dependency that's pulling in `google-gax` / old `grpc`.

   Run this in your project root:

   ```bash
   npm ls google-gax
   npm ls grpc
   ```

   This shows which package is bringing it in (very often `@google-cloud/some-service`, `@google-cloud/firestore`, old Dialogflow SDK, Speech-to-Text, etc.).

   Then either:

   - Update that package to its latest version (`npm update @google-cloud/...` or edit package.json).
   - Or — if it's not actually used — remove it completely.

2. **Force modern grpc-js (most common workaround in 2024–2026)**

   Many older Google libraries can be told to use `@grpc/grpc-js` instead of the ancient `grpc`.

   Add this to the very top of your `package.json` (or create resolutions if using yarn):

   ```JSON
   "resolutions": {
     "grpc": "@grpc/grpc-js",
     "google-gax/grpc": "@grpc/grpc-js"
   }
   ```

   Then delete `node_modules` + lockfile and reinstall:

   ```Bash
   rm -rf node_modules package-lock.json   # or yarn.lock
   npm install                             # or yarn
   ```

   If using **yarn v1** classic, use the resolutions field as above.
   If using **yarn berry (v2+)** or **pnpm**, the syntax is slightly different — let me know which package manager you're using.

3. **Downgrade Node.js (temporary band-aid only — not recommended long-term)**

   ```Bash
   nvm install 16
   nvm use 16
   rm -rf node_modules package-lock.json
   npm install
   ```

   Node 16 or 18 sometimes still fails with old grpc@1.13, but Node 14 or 12 usually works. Still — this is just kicking the can; you should upgrade dependencies eventually.

4. **If it's a Google Cloud library you need**

Update to the latest `@google-cloud/*` packages — most of them switched to `@grpc/grpc-js` years ago.

Example:

```bash
npm install @google-cloud/firestore@latest @google-cloud/storage@latest
```

Or just run:

```bash
npx npm-check-updates -u
npm install
```

Then fix any breaking changes.

Summary – What to Try First (5 minutes)

```bash
# See what's pulling in the bad deps
npm ls google-gax
npm ls grpc

# Force modern grpc
# Add to package.json → "resolutions": { "grpc": "@grpc/grpc-js", "google-gax

rm -rf node_modules package-lock.json
npm cache clean --force
npm install
```

If that doesn't resolve it, paste the output of `npm ls grpc + npm ls google-gax` here (or the name of the package that depends on google-gax), and I can give a more precise fix.

You're hitting a very common problem when old Google Cloud example projects or boilerplates meet modern Node — it's fixable in almost all cases without downgrading Node.