# Honey Lens Symbolic Market Tracker — C++ Setup Appendix

*(For publication bundle, Aug 14 release)*

## A. Get the required json.hpp header (nlohmann/json)

**This project uses a single-header JSON library. No installer is needed.**

### Option 1 — Direct single‑header download (quickest)

1. Create a folder named nlohmann inside your project directory.
2. Download the header and save it as:

* <your\_project>/nlohmann/json.hpp

1. In your C++ file, include it with:

* #include "nlohmann/json.hpp"

### Option 2 — From the repository’s single\_include folder

1. Download or clone the nlohmann/json repository.
2. Copy the entire folder:

* json-develop/single\_include/nlohmann
* into your project so you have:
* <your\_project>/nlohmann/json.hpp

1. Use the same include line as above.

**Ultra‑minimal fallback:** Place json.hpp in the same folder as symbolic\_tracker.cpp and include with #include "json.hpp" (no subfolder).

## B. Project layout (minimal)

<your\_project>/  
 symbolic\_tracker.cpp  
 nlohmann/  
 json.hpp

## C. Quick‑Start Build

### Windows — Visual Studio (MSVC)

1. **Create** a new C++ Console App project.
2. **Add** symbolic\_tracker.cpp to *Source Files*.
3. **Add** nlohmann/json.hpp to the project (or just ensure it exists in the project folder as above).
4. Set **Language Standard**: *Project → Properties → C/C++ → Language → C++17* (or later).
5. **Build → Rebuild Solution**.

**If you see a** ``\*\* safety warning (C4996)\*\*

* Easiest fix (recommended): use secure CRT in the helper function:
* std::string get\_today\_date() {  
   std::time\_t t = std::time(nullptr);  
   std::tm tm{};  
   localtime\_s(&tm, &t); // MSVC secure variant  
   char buf[11];  
   std::strftime(buf, sizeof(buf), "%Y-%m-%d", &tm);  
   return std::string(buf);  
  }
* Or disable “treat warnings as errors”: *Project → Properties → C/C++ → General → Treat Warnings As Errors = No (/WX-)*.
* Or define \_CRT\_SECURE\_NO\_WARNINGS in preprocessor definitions.

### Windows/macOS/Linux — g++ / clang++

From a terminal in your project directory:

g++ -std=c++17 symbolic\_tracker.cpp -o symbolic\_tracker  
./symbolic\_tracker

On very old GCC versions, if <filesystem> link errors appear, add -lstdc++fs.

## D. Using the Tracker (runtime flow)

1. Run the program.
2. For each symbol, enter: **Last Price, Daily Change ($), Percent Change (%), Cost Basis, Quantity**.
3. Type DONE when finished.
4. The program prints a **session summary table** with:
   * Value, P/L ($, %), and **Symbolic Phase** (*Inhale / Exhale / Pause*).
5. (Optional) If file logging is enabled in your build, a JSON log will be written to symbolic\_logs/DATE.json.

## E. Common Errors & Fixes

* **E1696 / C1083**: *cannot open include file* nlohmann/json.hpp → ensure the header is at <your\_project>/nlohmann/json.hpp or use the ultra‑minimal fallback and #include "json.hpp".
* **C4996**: 'localtime' may be unsafe → use localtime\_s (MSVC), or disable warnings‑as‑errors, or define \_CRT\_SECURE\_NO\_WARNINGS.
* **Filesystem link errors (older GCC)** → compile with -std=c++17 and add -lstdc++fs.

## F. Minimal Include Example (copy/paste)

#include <iostream>  
#include <iomanip>  
#include <ctime>  
#include "nlohmann/json.hpp" // or "json.hpp" if placed next to the .cpp

This appendix is designed to be dropped into the publication bundle so readers can reproduce the tracker locally with minimal friction.