

## QuotePilot Development Timeline (Updated)

2025-11-14

- Selected and purchased new development laptop.
- Installed Visual Studio Code and Python and confirmed environment setup.
- Installed Bitwarden and began full password overhaul.
- Set up GitHub access and verified coding environment functionality.

2025-11-15

- Registered quotepilothq.com domain.
- Evaluated naming strategy and domain availability.
- Confirmed environment running Python test scripts correctly.

2025-11-17

- Searched for naming conflicts for QuotePilot.
- Confirmed readiness for LLC formation and trademark filing.
- Outlined long term AI driven rules engine vision for instrumentation.

2025-11-18

- Formed QuotePilot LLC officially.
- Created operating agreement, banking resolution, and ownership ledger.
- Obtained EIN for QuotePilot LLC.
- Set up QuotePilot project folder and Backend structure in Visual Studio Code.
- Created first test scripts and confirmed OpenAI connectivity.
- Created full QPSAH200S part number definition and saved rules for engine development.
- Filed QuotePilot trademark application with USPTO and submitted payment.

2025-11-19

- Built full DP transmitter engine with parsing, validation, and pricing.

- Implemented structured error handling for invalid codes.
- Created API wrapper functions and improved structured outputs.
- Installed FastAPI and Uvicorn and built working api.py.
- Added GET health endpoint and POST quote endpoint.
- Served API locally and verified functionality through browser and FastAPI docs.
- Built HTML UI (quote\_ui.html) and added GET ui endpoint.
- Confirmed end-to-end functionality from browser to API to engine.
- Created external Python client for testing API responses.

2025-11-20 (Today)

- Created dedicated PartNumberEngine folder for scalable multi-product support.
- Added base\_engine.py defining the abstract engine structure.
- Built QPSAH200SEngine engine wrapper separating DP logic cleanly.
- Implemented engine registry for multi-model support.
- Updated api.py to dynamically load engines using request.model.
- Added model support to the API request.
- Updated HTML UI to send model and added model dropdown selector.
- Confirmed full end-to-end functionality with new architecture.