

FrameForge PRO

Technical overview for external reviewers

- PHP 8.1 (no framework) + SQLite + vanilla JS + Three.js r128
- STEP built server-side with FreeCAD solid booleans
- Verified by calculation: 1620-variant grid, 3000 random boxes, 32 real STEP boxes
- Built by a mechanic with no coding background together with AI assistants

Contents

- | | |
|--------------------------|------------------------|
| 1. What this is | 5. How it was verified |
| 2. Stack and constraints | 6. Known open issues |
| 3. Module map | 7. Auth model |
| 4. Design decisions | |

1 What this is

FrameForge PRO is a browser-based parametric CAD tool for aluminium enclosures manufactured on CNC mills or laser cutters. It goes from a few numbers (inner width/height/depth, fastener size) to a six-panel box with corner/support hardware, partitions and clamp bars, then exports production files.

What we would like from you: read the code critically (architecture, anti-patterns, anything you would flag in a review), then build a box end-to-end and tell us where the UX confused you.

2 Stack and constraints

Layer	Choice
Backend	PHP 8.1, no framework; SQLite (PDO) for projects.
Frontend	Vanilla JavaScript, no build step; HTML5 canvas for 2D, Three.js r128 for 3D.
STEP export	Server-side, headless FreeCAD (freecadcmd): true solid booleans (box minus cylinder/cone), exact analytic arcs.
Hosting	Shared Ubuntu 22.04 VPS, nginx + PHP-FPM; no containers, no CI.
i18n	4 languages (CS/EN/DE/RU), flat key-value dictionaries, client-side.

Browsers down to Chrome 109 / Windows 7 are supported on purpose (the owner's workshop machine).

3 Module map

File	Responsibility
cad_auto.js	Math core: six panels, corner/support cubes, partitions, auto cube size.
cad_math.js	Anchoring/unanchoring of elements, hole geometry.
_cube_pattern.php	Single source of the cube hole layout for all PHP generators (JS twin in cad_auto.js); canonical signature over the 24 cube rotations.
ff_bom.js	Fastener list and clamp-bar deflection check.
viewer_core.js / viewer_utils.js	Three.js assembly and the volume-based production check (bolt-to-cube match, collisions, edge webs).
gen_step.php	STEP assembly via FreeCAD; writes README, manifest and fastener list.
gen_cubes.php / gen_mount_cubes.php	Fusion 360 scripts for cubes (spaced apart; participant bodies).
_hole_finish.php	Shared countersink/counterbore geometry for all exporters.
ui_hub.js	Single delegated event router for the UI.

4 Design decisions

Coordinates

Element coordinates are always stored panel-absolute. Reference-point ($\Delta X/\Delta Y$) conversion is display-only, in one function, so there are never two sources of truth.

One computation, one place

The most frequent bug class was the same calculation reimplemented in several places that drifted apart (bolt counts, cube counts, edge clearance; later the cube hole layout and tap diameters). The cube layout is now shared in one PHP file and mirrored in JS, and a test confirms PHP and JS agree (238 of 238 combinations).

Why solid booleans for STEP

Extruding a flat profile in Fusion produced arcs faceted into micro-segments at the factory. The server builds each part as a box minus cylinders/cones in OpenCASCADE, so arcs remain exact circles in STEP.

Threads

Threads are cut only in the cubes (tap-drill $\varnothing 2.5$ / 3.3 / 4.2 mm for M3/M4/M5). Physically there are four distinct cube types, whatever their orientation in the box.

5 How it was verified

- Grid of 1620 box variants (M3/M4/M5, sheet 2/3/5 mm, ears, three sizes, all partition types): zero errors, except crossed partitions, which warn by design.
- 3000 random boxes with a different thickness on every panel (1.5–6 mm): zero collisions, zero missed bolts, zero overlapping holes in normal-size boxes.
- 32 of 32 generated STEP boxes re-opened with OpenCASCADE and checked geometrically (hole count, diameters, positions).

- Browser scenarios: 40 boxes, every bolt matches its cube; zero JS errors.
- Owner's own factory-proven Fusion scripts were used as the reference for cube geometry.

Honest limit: the real FreeCAD run happens on the owner's server ("Reference box" in the admin panel must show 7/7). A full STEP production run at the factory is still pending.

6 Known open issues

- Crossed transverse + longitudinal partitions only warn; the generator does not split one of them yet.
- Undo does not fully cover a cube drag across panels (the pre-export 3D check catches the mismatch).
- The "edge web $\geq$ sheet thickness" rule is conservative for 4-6 mm sheets.
- Side channels of a manually placed cube are not drawn in 3D.
- Oblique (slanted) ventilation is not implemented.
- No automated test suite in the repository; checks are scripted harnesses kept outside it. A small harness around the pure-math files would be the best contribution.

7 Auth model

Single operator, master password with timing-safe comparisons, CSRF tokens on state-changing calls, per-project PIN against accidental overwrite. No user accounts or roles by design. No credentials or keys in the code base; secrets are generated at first run.

This document was prepared by Claude (Anthropic) at the project owner's request for external review. Questions can be sent through the owner.

FrameForge PRO, parametric CAD, PHP, vanilla JavaScript, Three.js, FreeCAD, STEP export, OpenCASCADE, code review, AI-assisted development