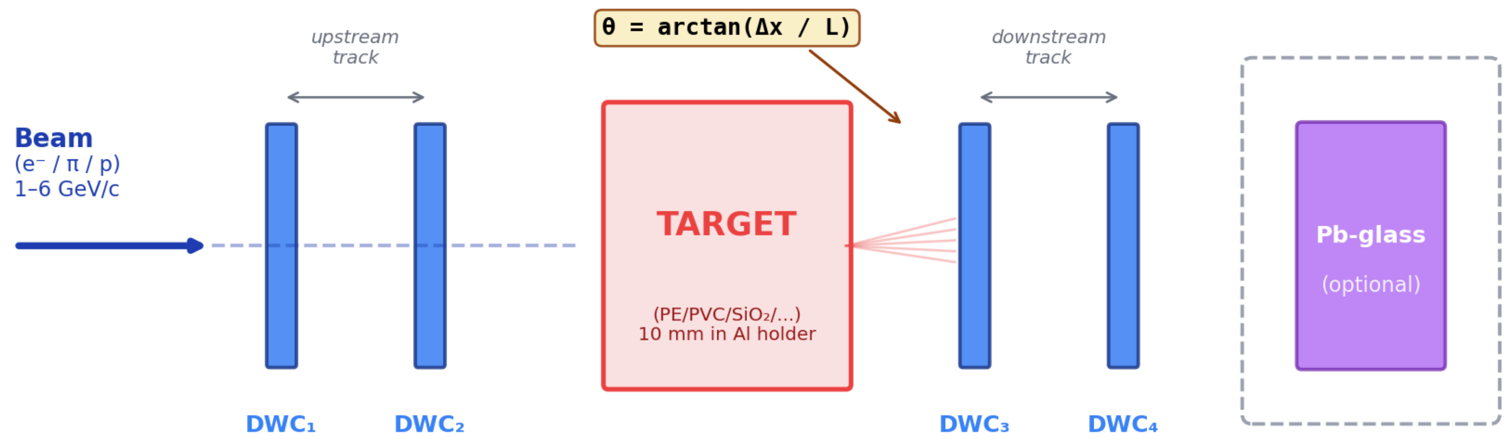
Technical Deep Dive

What the code actually does — for project owners

7 source files · 2 CI pipelines · 3 analysis scripts · 11 materials

Geant4 — Detector Geometry

BeamScan — Experimental Setup (facility-agnostic)



Component	Material	Size	Purpose
DWCs (×4)	Ar/CO_2 80/20 gas (10 mm gap)	10 × 10 cm^2	Measure particle direction before/after target
Target	Configurable via macro	10 × 10 cm^2, variable thickness	Material under study
Calorimeter	PbO lead glass	15 × 15 × 30 cm^3	Energy measurement (optional extension)

Geant4 — Physics List & Per-Event Data

Physics List: FTFP_BERT + option4

FTFP_BERT: Fritiof + Bertini cascade

- Handles hadronic interactions (p, π, n)
- Nuclear elastic + inelastic scattering
- *This is why G4 gives 12% more than Highland*

EM option4: most accurate MCS model

- G4UrbanMscModel for electrons
- G4WentzelVIModel for hadrons
- Step limiter ensures small steps in

What happens per event

1. `ParticleGun` fires e^- at $z = -80$ cm
2. Particle traverses DWC1 → DWC2 *(SteppingAction records first hit per plane)*
3. Particle enters **TARGET** → MCS happens
4. Particle exits → DWC3 → DWC4
5. Particle hits Calorimeter → energy deposited
6. `EventAction` computes all angles + momenta

Geant4 — The 9-Column Ntuple

Column	Unit	How computed	Used for
`theta3D_mrad`	mrad	`acos(inDir · outDir)`	Full 3D scattering angle
`thetaX_mrad`	mrad	`atan2` difference in x-z plane	**Primary classification observable**
`thetaY_mrad`	mrad	`atan2` difference in y-z plane	Cross-check / 2D analysis
`pIn_GeV`	GeV	Momentum at DWC2 (before target)	Beam energy verification
`pOut_GeV`	GeV	Momentum at DWC3 (after target)	Energy loss measurement
`deltaP_MeV`	MeV	pIn − pOut	dE/dx spectroscopy
`xTarget_mm`	mm	Interpolated x position at target plane	Beam profile
`yTarget_mm`	mm	Interpolated y position at target plane	Beam profile
`caloEdep_MeV`	MeV	Total energy deposited in lead glass	Particle ID (future)

> **9 columns per event** — enough for MCS classification AND momentum-loss spectroscopy. Students can mine this data for multiple physics analyses beyond the primary classification.

Geant4 — Multi-Threading & Output Format

MT-Mode Architecture

Geant4 11.3.2 auto-detects CPU cores. GitHub Actions runner: **4 cores → 4 threads**.

Each thread writes its own CSV file:

```
events_nt_beamscan_t0.csv
events_nt_beamscan_t1.csv
events_nt_beamscan_t2.csv
events_nt_beamscan_t3.csv
```

`ActionInitialization.cc` creates per-thread copies of `RunAction` + `EventAction` + `SteppingAction`.

WCSV Format (not standard CSV!)

Geant4 writes `#class tools::wcsv::ntuple` headers with metadata, **not a plain CSV**.

Our parser (`analyze_geant4.py`) handles:

- `#separator` line → detects delimiter
- `#column` lines → extracts column names
- Data lines → splits and zips into dicts

Why this matters:

`pandas.read_csv()` **fails** on Geant4

Analysis Pipeline — From Raw Events to Classification

1. Raw CSV
θx, θy, θ3D,
pIn, pOut, ΔP,
x, y, Ecalo
per event

→

2. Core Cut
Remove tails:
keep central 95%
(2.5→97.5 pctl)
Mitigate nuclear
scattering outliers

→

3. Gaussian Fit
σ_core = std(core)
μ_core =
mean(core)
This is the G4
"measured" σ

→

4. Highland Match
Load
predictions.csv
Match
material_pGeV
Compute ratio:
G4_σ /
Highland_θ_0

→

5. Outputs
Bar plot
Comparison table
CSV data
SUMMARY.md

Why the Core Cut Matters

MCS produces a **Gaussian core + non-Gaussian tails**. The tails come from: (a) nuclear elastic scattering (large single kicks), (b) nuclear inelastic events (secondaries), (c) delta rays. Highland predicts only the Gaussian core, so we cut the tails for a fair comparison.

Highland Calculator — The Analytic Engine

The Formula (PDG 2024)

$$\theta_0 = \frac{13.6\ \mathrm{MeV}}{p\beta c}\sqrt{\frac{x}{X_0}}\left[1 + 0.038\ln\frac{x}{X_0}\right]$$

`compute_predictions()` does:

1. Look up X_0, ρ from `MATERIALS_DB`
2. Or accept **student-provided** X_0/ρ in YAML *(custom materials!)*
3. Compute θ_0 via Highland formula
4. Estimate $dE/dx \approx 2\ \mathrm{MeV}/(\mathrm{g/cm^2}) \times \rho \times x$
5. Store result: θ_0, dE, X_0, ρ, p, thickness
6. For all material pairs:

Output Files

`predictions.csv` — 11 materials × 2 momenta = 22 rows

Columns: name, g4name, category, X_0, ρ, thickness, p, θ_0, dE

`distributions.png` — Gaussian curves overlaid for all materials

`classification.png` — 2D plot: θ_0(3 GeV) vs θ_0(6 GeV) with 1/p line and cluster annotations

`SUMMARY.md` — Table + pair separations + N_events

CI/CD — Two Pipelines, Two Purposes

⚡ Highland CI (~30 sec)

Triggers: PR touching `requests/*.yaml`, push to main, or manual dispatch

Steps:

1. Checkout repo
2. `pip install pyyaml jsonschema matplotlib`
3. Validate YAML against JSON schema
4. Run `highland_calculator.py`
5. Upload `predictions/` as artifact
6. If PR: **post comment with plots +**

⚛ Geant4 CI (~20 min)

Triggers: manual dispatch only *(too expensive for every PR)*

Steps:

1. Checkout repo
2. Install Miniforge (conda)
3. `conda install geant4` (11.3.2) + cmake, gcc, matplotlib, scipy
4. `cmake + make beamscan`
5. `generate_macros.py` → 22 `.mac` files
6. `run_all.sh` → runs all 22 simulations

Macro Generation & Scattering Computation

`generate_macros.py`

Reads student YAML → writes one `.mac` per material × momentum:

```
# Auto-generated: PE_3.0GeV_10.0mm.mac
/run/initialize
/gun/particle e-
/gun/energy 3.0 GeV
/beamscan/target/material G4_POLYETHYLENE
/beamscan/target/thickness 10.0 mm
/beamscan/output/filename \
  geant4_output/PE_3.0GeV_10.0mm/events
/run/beamOn 2000
```

Plus `run_all.sh`:

```
for f in macros_auto/*.mac; do
```

The Scattering Angle Computation

`SteppingAction` records first hit per DWC → position + momentum at each plane.

`EventAction::EndOfEventAction` computes:

```
inDir  = (pos[DWC2] − pos[DWC1]).unit()
outDir = (pos[DWC4] − pos[DWC3]).unit()

θ_3D = acos(inDir · outDir)
θ_x  = atan2(out.x, out.z) − atan2(in.x, in.z)
θ_y  = atan2(out.y, out.z) − atan2(in.y, in.z)
```

Only events hitting **all 4 DWCs** are kept (natural acceptance cut).

Bugs Found & Fixed — Engineering Story

Severity	Bug	Root Cause	Fix
CRITICAL	**X_0 Values Wrong (4 materials)**	PVC, $CaCO_3$, Al_2O_3, Fe_2O_3 had X_0 in g/cm² instead of cm. Highland overpredicted PVC (physically impossible).	Recomputed using Tsai formula + PDG cross-checks
HIGH	**PET/PE Substring Collision**	`"PE" in "PET_3.0GeV"` → `True` in Python. Highland matching grabbed PE's value for PET.	`m.startswith(material + "_")` instead of `in`
MEDIUM	**Workflow Push Race Condition**	20-min Geant4 run + concurrent push → rejected.	`git pull --rebase` before `git push`
REJECTED	**ChatGPT Review — 8 Regressions**	Reviewed old snapshot. Deleted `ActionInitialization` (breaks MT), removed wcsv parser, reverted to hardcoded filenames, used wrong PE X_0.	All changes rejected after analysis

🟢 **Every bug has a commit, a rationale, and a verification.** This is engineering discipline that BL4S reviewers will notice — it shows the team understands what their code does.

Request Schema — What Students Can Configure

Full YAML Structure

```yaml
author: "Name"                # required
description: "Research Q"   # required
materials:                   # 1+ entries
  - name: PE                 # display name
    geant4_name: G4_POLYETHYLENE
    thickness_mm: 10         # 1-100 mm
    # Optional: override for custom materials
    X0_cm: 47.9
    rho: 0.94
beam:
  particle: e-      # e-, e+, pi+, pi-
  momenta_GeV: [3.0, 6.0]  # list
num_events: 10000  # per configuration
```

Validated by `schemas/request.schema.json` (Draft 2020-12).

Named Examples in Repo

Valentina — `valentina_pvc_detection.yaml`
PVC vs PE → recycling QC

Tomás — `tomas_thickness_study.yaml`
PE at 5, 10, 20 mm → test $\sqrt{(x/X_0)}$ scaling

Lucía — `lucia_heritage_study.yaml`
SiO_2, $CaCO_3$, Al_2O_3, Fe_2O_3 → heritage materials

Sofía — `sofia_custom_material_example.yaml`
Student provides own X_0 and ρ → custom

End-to-End

Student types → gets publication-ready physics plots back in

📝 YAML → 🔀 PR → 🤖 CI → 📊 Plots → ✅ Merge → ⚛️ Geant4 → 🌐 Website

¡La física fundamental es para todos!