SBI HACKATHON 2026: CONTRIBUTOR TRACK

LOCATION
Grenoble, France

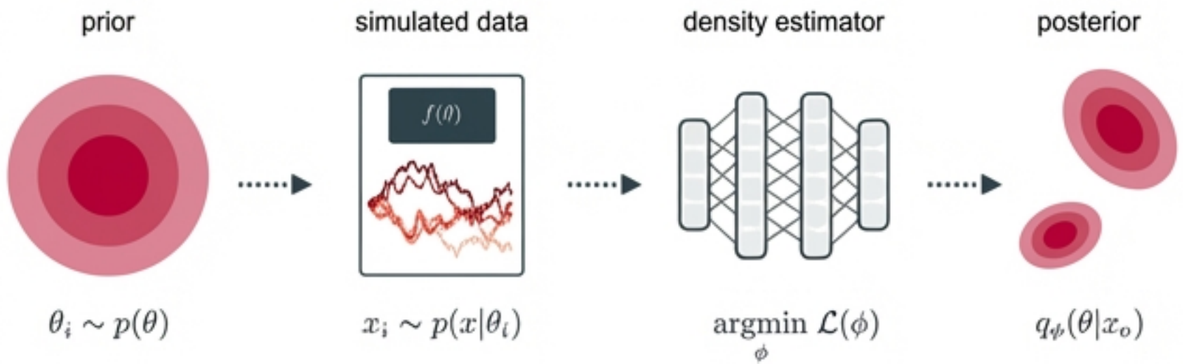
DATE
January 21-23, 2026

FOCUS
Architecture & Roadmap

> FROM RESEARCH CODE TO USER-ORIENTED PACKAGE

High-Level Organization: Mapping Science to Source

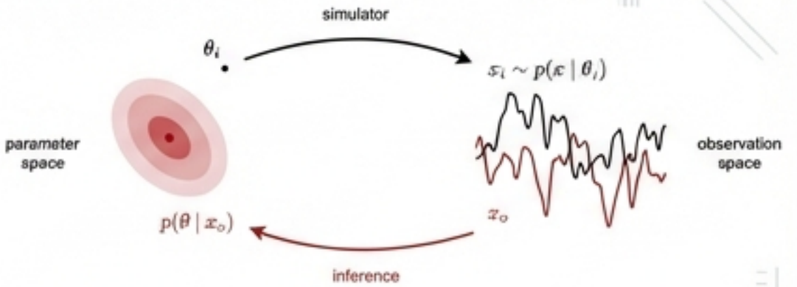
THE MENTAL MODEL (Science)



THE REPOSITORY (Code)

```
sbi/
├── inference/     <-- The "Brains" (Trainers, Algorithms)
│   ├── trainers/
│   └── posteriors/
├── neural_nets/  <-- The "Engine" (Density Estimators)
│   ├── flow/
│   └── embedding_nets/
├── samplers/     <-- Support (MCMC wrappers)
├── utils/        <-- Support (Metrics, user diagnostics)
└── diagnostics/
```

The Life of an Inference Task

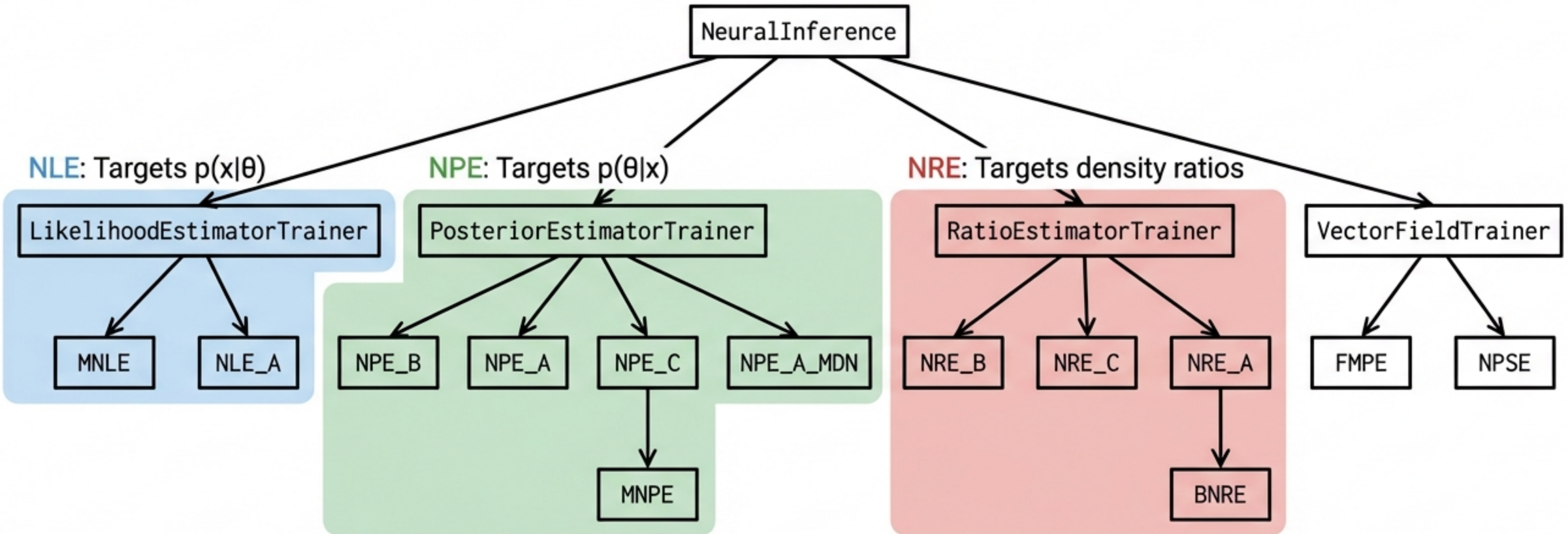


Prior / Simulator → `append_simulations(theta, x)` → Data Ingest → `train()` → Optimization → `build_posterior()`

Design Pattern: Trainer-Posterior Separation

- **Trainers**: Handle optimization loop and simulation management.
- **Posteriors**: Handle sampling, `log_prob` evaluation, and device management.
- **Why?** Allows swapping sampling backends (e.g., Rejection vs. MCMC) after training without retraining the network.

Orchestrating Inference: The Trainer Hierarchy



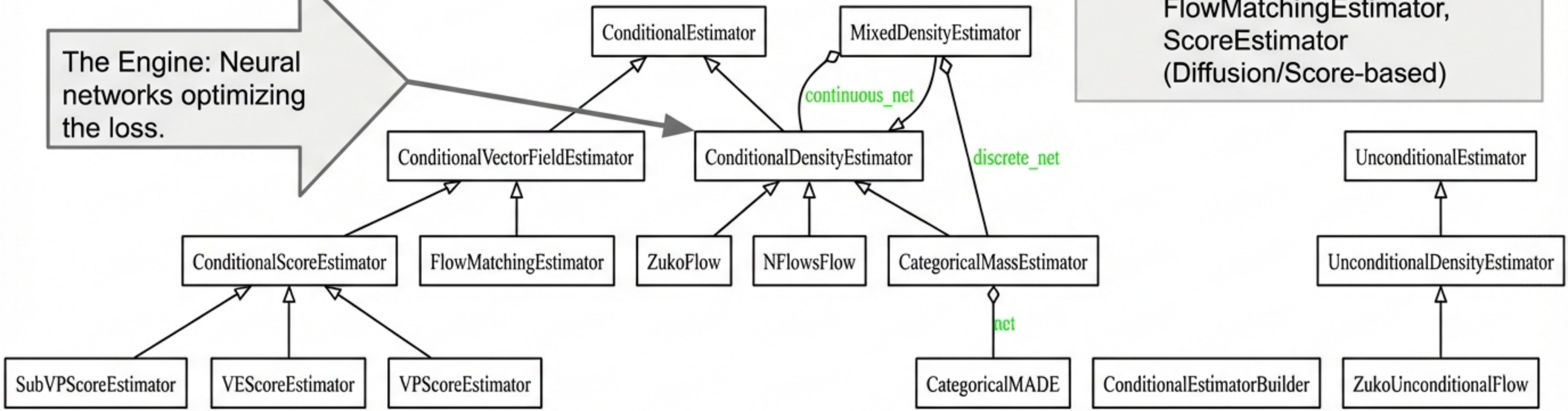
Root: `NeuralInference` handles the common training loop. Leaf nodes (e.g., `NPE_C`) represent specific algorithm variants.

Powering Inference: Density Estimators

Key Components

- **Flows:** NFlowsFlow, ZukoFlow (Normalizing Flows)
- **Next-Gen:** FlowMatchingEstimator, ScoreEstimator (Diffusion/Score-based)

The Engine: Neural networks optimizing the loss.



Evolution Note: Moving away from factory functions (posterior_nn) toward direct class instantiation (See Issue #1659)

The User Interface: Posterior Classes

Direct / Amortized

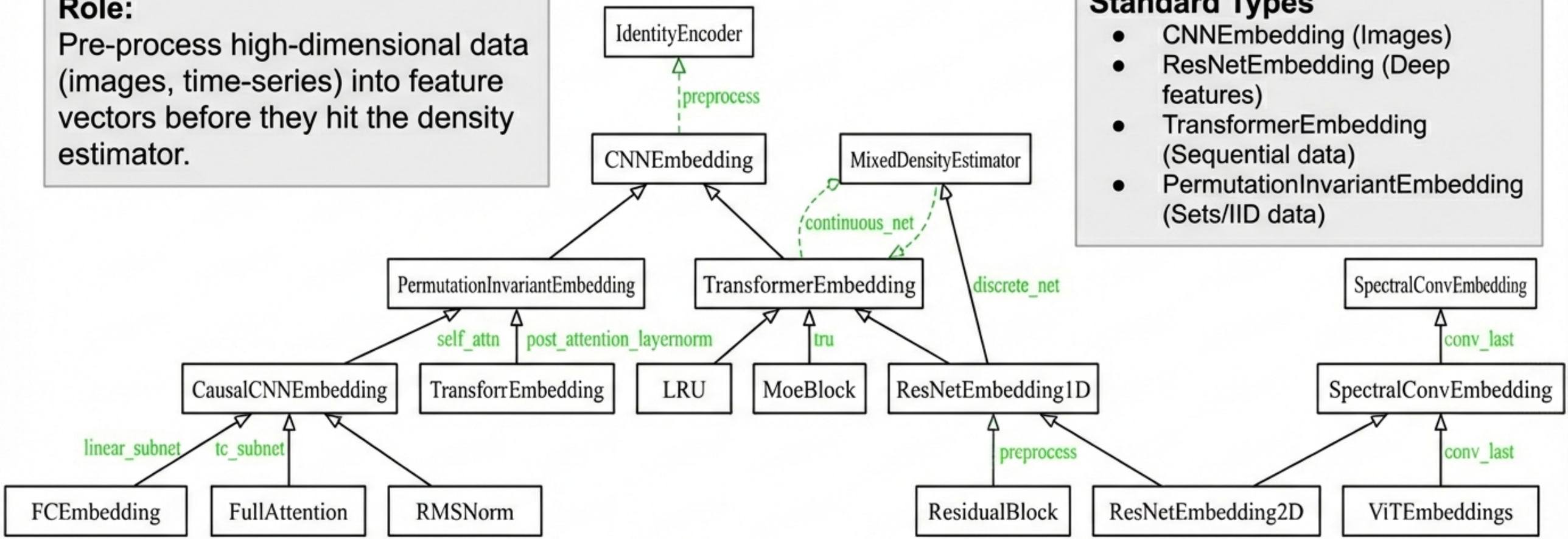
- **Class:** DirectPosterior
- **Use Case:** NPE methods
- **Feature:** Instant sampling via the normalizing flow's inverse pass.

Sampling-Based (MCMC/VI)

- **Classes:** MCMCPosterior, VIPosterior, RejectionPosterior
- **Use Case:** NLE / NRE methods
- **Mechanism:** Potential Functions
  Potential = log_likelihood + log_prior
  (Allows flexible sampler choices via sbi.samplers)

Handling Complex Data: Embedding Networks

Role:
Pre-process high-dimensional data (images, time-series) into feature vectors before they hit the density estimator.



Standard Types

- CNNEmbedding (Images)
- ResNetEmbedding (Deep features)
- TransformerEmbedding (Sequential data)
- PermutationInvariantEmbedding (Sets/IID data)

Implementation Note

Must inherit from nn.Module and is trained end-to-end with the inference network.

The API Contract: Essential Methods

Trainer Contract (sbi.inference)

```
def append_simulations(theta, x): # Add training data
def train(): # Execute training loop
def build_posterior(): # Return posterior object
```

Posterior Contract (sbi.inference.posteriors)

```
def sample(shape, x): # Generate parameters
def log_prob(theta, x): # Evaluate density
def map(x): # Find Maximum A Posteriori
```

Density Estimator Contract (sbi.neural_nets)

```
def loss(inputs, condition): # Calculate training objective
```

Development Setup: Getting Ready to Code

Part 2

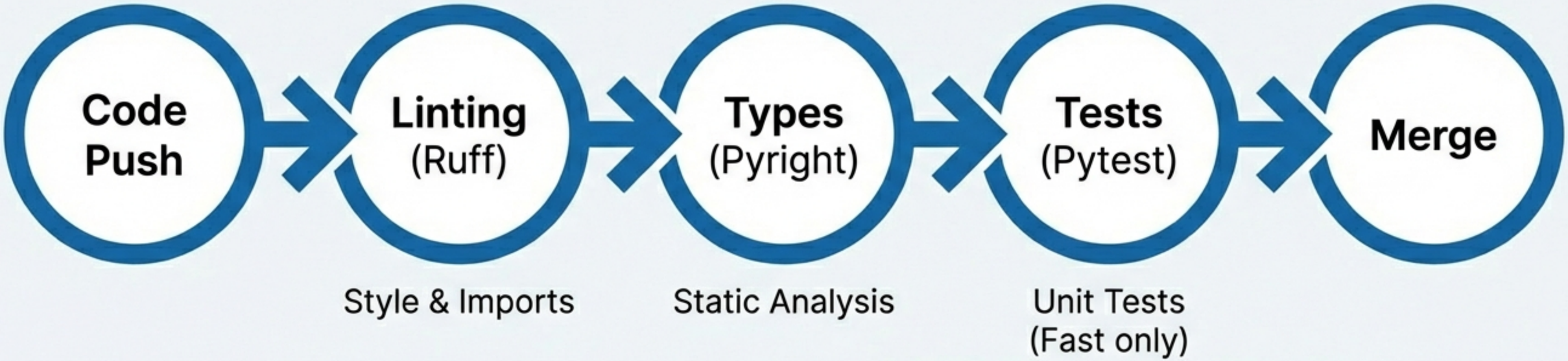
```
$ git clone https://github.com/username/sbi.git
$ cd sbi

$ # CRITICAL: Install in editable mode with dev dependencies
$ uv pip install -e ".[dev]"

 # Install hooks to catch style issues early
$ pre-commit install
```

Pro Tip: We use 'uv' for modern, fast Python package management.

The CI/CD Pipeline: What Happens When You Push?



Note: CI runs fast tests only. GPU and slow tests are separate workflows.

Testing Best Practices

```
pytest -m "not slow and not gpu"
```

Structure

Tests live in `tests/`. Use fixtures for common setup to reduce runtime.

Markers

Decorate heavy tests with:

`@pytest.mark.slow`
`@pytest.mark.gpu`

Performance

Check Mini-SBIBM for benchmarking new methods against standard baselines.

PR Etiquette: Getting Merged

Title

feat: Add flow matching estimator

Description

1. WHAT does it do?
2. WHY is it needed?
3. HOW does it work?
4. Fixes Issue #...

Pre-Flight Checklist

- ☑ Passes ruff & pyright
- ☑ Tests pass locally
- ☑ New functionality has new tests
- ☑ Documentation updated

Pick Your Challenge: Hackathon Projects

Good First Issues

- Docstring cleanup (#1534)
- Expose FMPE/NPSE hyperparameters (#1284)
- GPU-compatible diagnostics (#1160)

Documentation & Guides

- Hyperparameter tuning guide (#1346)
- Saving/Loading estimators guide (#1525)
- Pyro integration guide (#1517)

Advanced / Architecture

- Better flow matching training (#1445)
- Refactor NPE_A_MDN (#1408)
- Remove posterior_nn factory functions (#1659)

Ready? Let’s Build.

Next Steps

1. 60 min interactive discussion
2. Form working groups
3. Assign mentors

Docs: sbi.readthedocs.io | Discord: #hackathon-2025