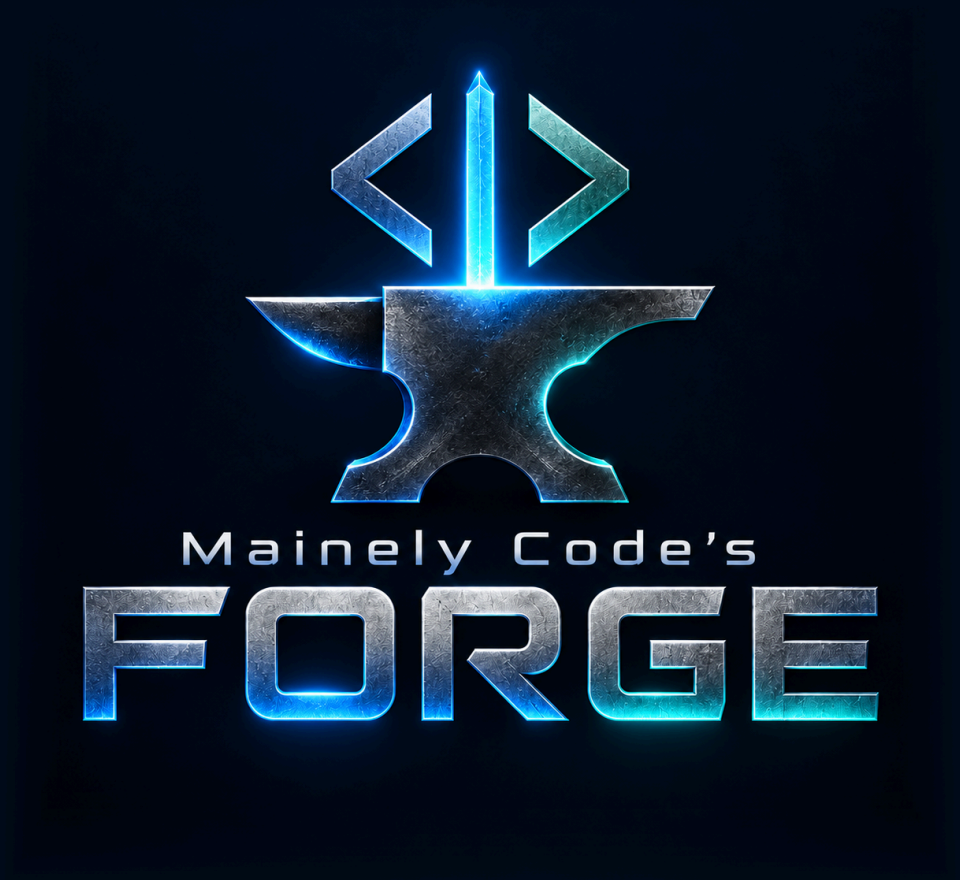
A MODEL ENGINEERING & AI RESEARCH WORKBENCH



Build the AI. Not just with it.

A planned standalone product for developing, modifying, and training specialist models—and exploring new AI technologies.

PLANNED PRODUCT

Development starting soon.

01 / THE VISION

A workbench for building AI itself.

Mainely Code's FORGE is a planned local-first model engineering and research workbench for developers, independent researchers, and small technical teams.

The direction is to bring data preparation, model development, training, experimentation, evaluation, and export into a coherent working environment. The initial audience is builders with high-end PCs and workstations who want to put their own computing power to work.

> Develop a specialist for a real task.
> Modify an existing model.
> Test an idea that does not fit a preset.

FOR DEVELOPERS

Build for a purpose.

Explore models tailored to a particular toolchain, workflow, domain, or application rather than treating every problem as a general chat task.

FOR RESEARCHERS

Make room for ideas.

Investigate custom components, training approaches, and new architectures in a workspace that leaves room for code-level experimentation.

FOR SMALL TEAMS

Keep the work usable.

Carry the data, configuration, findings, and artifacts forward so an experiment can become a practical next step—not a forgotten checkpoint.

THE PRODUCT DISTINCTION

Buildroom helps developers build software with AI.
FORGE is intended to help developers build the AI.

02 / PLANNED CAPABILITIES

Develop. Modify. Train. Then understand the result.

The proposed scope covers the model-development lifecycle. Guided workflows would provide a starting point, not a ceiling on technical control.

01 Data & project preparation

Organize datasets, inspect examples, document sources and permissions, and separate training material from evaluation data before a run begins.

Clear inputs. Traceable preparation.

02 Model inspection & modification

Explore supported model configurations and components, understand what an experiment changes, and preserve the original model as a reference.

Visible changes. Preserved baselines.

03 Specialization & training

Adapt supported models to focused tasks through methods such as adapter training and fine-tuning, with deeper training paths considered as the product matures.

Task-specific models. Explicit training choices.

04 Custom experiments

Bring model definitions, training code, and evaluation ideas into a research workflow. Explore new approaches, including training from scratch at a feasible scale.

Code-level access. Room for novel work.

05 Evaluation & comparison

Compare candidates against a baseline, inspect improvements and regressions, and retain unsuccessful experiments as part of the research record.

Evidence of change. Honest outcomes.

06 Export & practical handoff

Package supported checkpoints or adapters with their configuration, lineage, and evaluation findings, with the goal of making them usable outside FORGE.

Portable artifacts. Documented limitations.

These are areas of planned product direction, not a released feature list. Specific models, methods, formats, and workflows will be defined and validated during development.

03 / ROOM TO EXPERIMENT

A guided path.
An open research bench.

FORGE is intended to make established workflows more approachable without limiting ambitious work to the options in a wizard.

GUIDED MODEL DEVELOPMENT

Start with a clear goal.

Choose a supported starting model, prepare appropriate data, establish a baseline, and run a controlled specialization experiment.

The aim is to make the next decision understandable: what to train, what to measure, and what the outcome means.

OPEN-ENDED AI RESEARCH

Go beyond the recipe.

Explore custom model components, training objectives, evaluation approaches, and architectures through a code-accessible research environment.

The ambition extends beyond tuning existing models to helping builders investigate genuinely new AI technologies.

Approachable defaults.
Deep technical access.
Space for work no preset can anticipate.

ILLUSTRATIVE PROJECT DIRECTIONS

A toolchain specialist

Investigate a model focused on a defined coding or IT workflow.

A compact task model

Explore a classifier or retrieval model for a narrow application.

An architecture experiment

Test a new component or training idea against a controlled baseline.

Examples illustrate the intended breadth. They do not represent completed FORGE projects or guaranteed launch support.

04 / THE INTENDED WORKFLOW

From a question to a model you can use.

A complete journey is the goal—not just a button that starts training and leaves the developer to make sense of the output.

01	**Define**	Describe the task, the hypothesis, the intended use, and the evidence that would demonstrate a useful result.
02	**Prepare**	Choose a starting model or custom definition, review the data, and check the proposed workload against available hardware.
03	**Baseline**	Evaluate the starting point before making changes. Preserve the comparison conditions and hold back appropriate test data.
04	**Experiment**	Run the training or modification, record the configuration and environment, and retain progress and checkpoint information.
05	**Compare**	Inspect what improved, what regressed, what remains uncertain, and how the candidate's resource demands changed.
06	**Package**	Export a supported artifact with its findings and limitations. Check that it loads in the intended compatible runtime before handoff.

> "Training finished" should never be mistaken for "the model improved."

THE RESEARCH RECORD

The planned record would connect the model revision, data snapshot, configuration, environment, checkpoints, evaluations, and exported artifact. A failed hypothesis should remain useful evidence.

Conceptual workflow. Exact execution, recovery, and export behavior will be established during development.

05 / HARDWARE & LOCAL CONTROL

Start with the machine you own.

High-end developer PCs and GPU workstations are the initial audience. FORGE's direction is to help builders make deliberate use of their hardware—with clear limits, not blanket compatibility promises.

A hardware-aware starting point.

Understand the machine.

Assess the relevant accelerator, memory, storage, drivers, and execution environment for the selected workflow.

Qualify the experiment.

Check the actual model and training configuration rather than treating a successful chat session as a training-readiness test.

Explain the choices.

Surface constraints and possible adjustments before committing to a long run. Make estimates and their uncertainty visible.

Planned readiness behavior—not a published hardware requirement or a performance guarantee.

Local-first by intent.

The core direction is to prepare data, run compatible experiments, and keep project artifacts on user-controlled hardware. Setup requirements, network behavior, and data-handling policies will be defined before release.

Additional compute by choice.

Optional private or remote compute may extend the workbench over time. The intention is explicit destination, authorization, and cost visibility—not silently moving a local project to an external service.

COMPATIBILITY WILL FOLLOW VALIDATION

Supported operating systems, GPU families, memory requirements, model sizes, multi-GPU configurations, and remote execution options have not yet been announced. Capability will be described for tested combinations, not inferred from a single hardware number.

06 / STANDALONE & CONNECTED

Standalone by design. Connected by choice.

FORGE is planned as its own product—not a Buildroom edition, a required subscription to another Mainely Code service, or a renamed inference runtime.

> The goal: train and export a supported model without requiring another Mainely Code product to use the result.

CONCEPTUAL HANDOFF · NOT A LIVE INTEGRATION

MODEL DEVELOPMENT

FORGE

Prepare, modify, train, evaluate, and package the model-development project.

→

MODEL EXECUTION

A compatible runtime

Load the exported artifact and run inference within its supported environment.

→

APPLICATION OUTCOME

The consuming product

Test the model in context and determine whether its output is fit for the task.

Interoperability as a goal.

Exports would carry the relevant model configuration, training lineage, evaluation findings, and applicable terms. Compatibility must be established for each format and runtime; a successful export alone is not proof of a successful deployment.

Clear roles across the family.

Potential Mainely Code integrations would preserve distinct responsibilities: FORGE for model development, PROVIDER for model execution, and Buildroom for software-engineering work and verification.

AI anywhere. Governed everywhere.
That company-wide direction means explicit permissions, visible handoffs, and evidence appropriate to each product—not treating model confidence as authority to act.

07 / THE DIRECTION AHEAD

More People should help build what's next.

OUR MISSION

> "Make state-of-the-art AI accessible, usable, and approachable for Everyone."

FORGE extends that mission from helping People use AI to helping them shape it. The ambition is practical: a workbench for specialist models, a place to test original ideas, and a clearer path from experimentation to something useful.

PLANNED PRODUCT DIRECTION

Development starting soon.

Current stage	General product direction and concept planning.
Next step	Begin development and validate the first end-to-end workflows.
To be announced	Release timing, supported models and hardware, distribution, pricing, and licensing.

Follow Mainely Code's progress

mainelycode.com

Mainely Code

AI anywhere. Governed everywhere.

This public vision brief describes a planned product, not currently available functionality. Scope and delivery plans may change as development and validation proceed. Examples are illustrative; no model-quality, benchmark, or hardware-performance claims are made.