Editor.Archiyou

A high-fidelity design of the Archiyou workspace, designed to support project-based workflows. It shows how users manage scripts, design files, and shared resources while maintaining clarity between tools, results, and workspaces.

1. Overall Interaction Model 4/30

- The editor provides a unified environment where parameters, code, and geometry coexist, allowing users to define reusable design capabilities rather than static outcomes. The editor supports exploratory and iterative design, enabling learning by doing across abstraction levels and aligns with parametric and system-based design thinking.

Experience rationale

- Supports exploratory, project-based work and learning by doing
- Ensures consistent logic by clearly separating tools, results, and project context
- Aligns with parametric design mental models and lesson-driven workflows

Parameters provide a structured control layer that exposes the key variables driving the design. Rather than modifying geometry directly, users adjust these parameters to guide rather than consult from.

The code editor exposes the underlying logic of the design, enabling transparency, learning, and advanced customization.

Show only meaningful results and exclude intermediate steps, while hiding complex noise without affecting the original.

The console provides real-time feedback and errors, making system state and execution visible during iteration.

Data inspection tools expose internal values and structures, supporting debugging and deeper understanding of the design.

The scene navigator reveals the structural composition of the model, enabling selection, inspection and validation.

The editor separates control (parameters), logic (code), and output (geometry) into distinct tool zones supporting complex workflows.

The 3D view reflects parameter and code changes immediately, reinforcing cause and effect relationships.

By starting from an existing structure, users focus on understanding and modification rather than setup.

2. Project Identity 4/30

- The project name and version establish a stable identity for the design, supporting versioning, iteration, and long-term traceability within a project.

Experience rationale

- Enables explicit versioning within project-based workflows
- Supports traceability between design changes and generated outputs
- Aligns with iterative and issue-driven design processes

Project metadata is edited independently from parameters and code, keeping high-level intent (name, version, capability, and geometry).

The project summary captures intent and context, helping others understand the purpose of the design beyond its geometry.

Key characteristics like difficulty, time, materials, and tools are documented in a structured format to support decision making and reuse.

Categories classify the project, improving discoverability and filtering library across projects and the community.

3. Menu 4/30

- The menu acts as a compact entry point for editor-level actions that are not part of the primary design flow.

Experience rationale

- Keeps advanced and infrequent actions accessible without cluttering the main interface
- Supports exploration and learning by exposing system-level capabilities
- Supports efficiency for experienced users through shortcuts and predictable structure

The menu acts as a compact entry point for editor-level actions that are not part of the primary design flow.

Not exercised in this small exploring, giving users control over when logs is evaluated rather than relying solely on automatic updates.

File-related actions are grouped together, including project management and import/export operations.

System-level resources are available without interfering with design work, supporting learning, troubleshooting, and exploration.

Advanced users can transition to external tooling while preserving the core as the single source of truth.

3. Onboarding 4/30

- The onboarding guides users through the editor by introducing core concepts and workflows step by step, without interrupting the design process.

Experience rationale

- Introduces key interface elements (parameters, objects, outputs) in context
- Supports learning by doing without forcing a linear tutorial
- Allows users to skip or resume onboarding based on experience level

Steps focus on mastering concepts rather than UI controls, supporting transferable understanding.

The onboarding is divided into clear steps, reflecting the core workflow from configuration to output.

The onboarding appears as a lightweight overlay, allowing users to continue exploring the editor while being guided.

Onboarding content is visually distinct from the editor interface, preventing confusion between guidance and controls.

3. Contextual AI assistance inside the editor 4/30

- The AI panel provides contextual, topic-aware assistance directly within the editor, helping users understand, extend, and reason about their designs while they work.

Experience rationale

- Reduces friction by keeping help within reach at the moment of need
- Facilitates guided exploration (parameters, code, geometry) into actionable guidance
- Scales from beginner support to expert acceleration without changing tools

The AI panel floats inside the editor, keeping assistance contextual rather than a separate documentation.

Suggested actions focus on common design tasks, keeping the barrier to asking the "right" question.

The AI helps users move from intent ("add parameters") "explain code") to concrete changes in the model.

Rather than starting completely, the AI explains decisions and supports learning alongside modification.

The panel is always available but never blocks the primary workflow.

3. Template library 4/30

- The template library provides curated starting points that help users explore structures, patterns, and parametric setups without starting from scratch.

Experience rationale

- Lowers the barrier to starting by offering concrete, editable examples
- Encourages exploration by showing what is possible, not prescribing outcomes
- Reinforces parametric thinking by exposing structure, parameters, and logic

Templates are presented as editable starting points, not finished results.

Categorization helps users locate by intent (e.g. learning, prototyping, manufacturing, fabrication).

Each template shows a visual preview combined with key dimensions to set expectations early.

Templates can be imported directly into the editor, enabling instant experimentation.

By starting from an existing structure, users focus on understanding and modification rather than setup.

3. Sharing & collaboration 4/30

- Sharing enables users to view, inspect, fork, and iterate, turning individual exploration into collaborative learning and co-creation.

Experience rationale

- Encourages knowledge sharing without forcing public publishing
- Lowers the barrier to collaboration by using familiar sharing patterns
- Supports learning through discussion, feedback, and pair iteration

Sharing is framed as an extension of learning, not a final publishing action.

The model keeps collaboration explicit by exposing specific people rather than broadcasting publicly.

Small visual tokens align with common collaboration patterns, reducing friction.

Users can start with private sharing before moving to broader publishing or community reuse.

Collaboration happens at the script level, reinforcing reuse and shared logic over static results.

3. Publishing scripts 4/30

- Publishing turns a personal script into a reusable capability that can be discovered, adapted, and extended by others.

Experience rationale

- Supports knowledge sharing and reuse by making scripts discoverable beyond direct collaborators
- Balances authorship, permissions, and visibility against broader publishing
- Supports a gradual transition from private exploration to community contribution

Publishing is presented as an explicit decision, separating experimentation from contribution.

Clear toggles allow authors to control how others can interact with the script (view, modify, share).

Making a script public enables discovery and reuse within the community without requiring direct sharing.

The generated link provides a persistent entry point for reuse, documentation, and discussion.