Algorithms & core behavior

- NL → Code: turn plain English app specs into full project scaffolds.
- Agentic workflow: plans architecture, files, routes, and DB schema (not just single-line completions).
- Uses LLMs (pluggable; examples mention OpenAI/Anthropic-style models) for prompt understanding and code generation.
- Supports debugging, refactoring suggestions, and explanatory help.
- Makes decisions/hypotheses about next steps (iterative, goal-directed).

Underlying infra & techniques (inferred)

- LLMs power generation and intent parsing.
- Produces conventional stacks (Node/Express, Python/Flask, React, etc.) — doesn’t invent new languages.
- Integrates with typical dev tools (package managers, ORMs, test runners).

Frontend UX & features

- Browser-based IDE with live preview and real-time collaborative editing (“multiplayer”).
- Chat-based prompt UI: “describe it” → watch it build.

- Generates frontends using common frameworks (React/Angular) and serves static assets automatically.

Backend & data

- Scaffolds backend code in common frameworks (Express, Flask, Next API routes).
- Built-in DB support (Replit DB) and environment/secrets management.
- Runs code in Replit’s cloud — no local server setup required.

Deployment & integration

- One-click / automatic deployment and hosting.
- Auto-wires frontend ▯ backend, routes, and DB connections.
- Secrets managed via environment variables / platform secrets.