



Roadmap - Gemini Context Extension

This document outlines planned features and tools for future versions of the Gemini Context Extension.

Current Version: 1.0.0

Included Tools:

-  Context Window Tracker
-  Cost Estimator

Version 2.0 - Context Optimization (Planned: Q1 2026)

Context Optimizer Tool

Problem: Users know they're using too much context but don't know how to reduce it.

Solution: Automated analysis and optimization suggestions.

Features:

- Identify unused or rarely-used MCP servers
- Detect redundant context files
- Suggest consolidation strategies
- Automatic context compression for long conversations
- Smart pruning recommendations

Technical Approach:

- Analyze tool invocation frequency from session history
- Parse context file content for redundancy detection
- Use heuristics to identify optimization opportunities
- Provide actionable "one-click" optimization actions

Estimated Complexity: High

Time Estimate: 4-6 weeks

Version 2.5 - Prompt Engineering Suite (Planned: Q2 2026)

Prompt Template Library

Problem: Developers repeatedly write similar prompts and don't have a systematic way to manage them.

Solution: Built-in library of proven prompt templates with variable substitution.

Features:

- 50+ pre-built prompt templates for common tasks
- Template variables and substitution
- Custom template creation and storage

- Template versioning and sharing
- Usage analytics for templates

Categories:

- Code review templates
- Documentation generation
- Test case creation
- Debugging assistance
- Architecture design
- Refactoring guides

Example Usage:

```
> Use the code-review template for src/api/users.ts
```

Estimated Complexity: Medium

Time Estimate: 3-4 weeks

Prompt Testing & Validation Tool

Problem: No way to test if prompt changes improve output quality.

Solution: A/B testing framework for prompts with quality metrics.

Features:

- Define test cases with expected outputs
- Run prompts against test suites
- Compare prompt variations (A/B testing)
- Quality scoring and metrics
- Regression detection for prompt changes
- Performance benchmarking

Technical Approach:

- Store test cases in `.gemini/tests/` directory
- Use similarity scoring for output comparison
- Track metrics: response time, token usage, accuracy
- Generate comparison reports

Estimated Complexity: High

Time Estimate: 5-6 weeks

Version 3.0 - Multi-Model Intelligence (Planned: Q3 2026)

Multi-Model Comparator

Problem: Hard to choose the right model for specific tasks or compare outputs.

Solution: Side-by-side comparison of multiple Gemini models.

Features:

- Send same prompt to multiple models simultaneously
- Compare outputs side-by-side
- Quality scoring and preference tracking
- Cost vs quality analysis
- Model recommendation engine

Technical Approach:

- Parallel API calls to different models
- Structured output format for comparison
- User preference learning over time
- Cost-benefit analysis per use case

Estimated Complexity: Medium

Time Estimate: 3-4 weeks

Model Performance Profiler

Problem: No visibility into response times, rate limits, or performance bottlenecks.

Solution: Comprehensive performance monitoring and profiling.

Features:

- Response time tracking per model
- Rate limit monitoring and warnings
- API error tracking and analysis
- Performance trends over time
- Bottleneck identification
- Optimal model selection for speed vs quality

Estimated Complexity: Medium

Time Estimate: 2-3 weeks

Version 3.5 - Session Intelligence (Planned: Q4 2026)

Session Analytics Dashboard

Problem: No historical insights into usage patterns or session metrics.

Solution: Comprehensive analytics for all Gemini CLI sessions.

Features:

- Session duration and token usage trends
- Cost analysis over time
- Most used tools and commands
- Context usage patterns
- Model switching analysis
- Export to CSV/JSON for external analysis

Technical Approach:

- Store session metadata in `.gemini/analytics/`

- Aggregate statistics across sessions
- Generate visual reports (ASCII art charts in terminal)
- Configurable retention policies

Estimated Complexity: Medium

Time Estimate: 3-4 weeks

Advanced Memory Manager

Problem: Basic `/memory` commands are limited; no smart organization or archiving.

Solution: Enhanced memory management with intelligent organization.

Features:

- Smart memory categorization (by topic, project, date)
- Memory search and filtering
- Automatic archiving of old memories
- Memory compression for token efficiency
- Memory templates and presets
- Cross-project memory sharing

Estimated Complexity: High

Time Estimate: 4-5 weeks

Version 4.0 - Workflow Automation (Planned: 2027)

Workflow Automation Builder

Problem: Complex multi-step workflows require manual orchestration.

Solution: Visual workflow builder with automation capabilities.

Features:

- Define multi-step workflows (e.g., "code review + test generation + documentation")
- Conditional logic and branching
- Loop constructs for batch processing
- Error handling and retries
- Workflow templates and sharing
- Scheduled execution

Technical Approach:

- JSON-based workflow definitions
- State machine execution engine
- Integration with Gemini tools and MCP servers
- Workflow visualization in terminal

Estimated Complexity: Very High

Time Estimate: 8-10 weeks

Output Post-Processor

Problem: Need to transform model outputs into different formats consistently.

Solution: Configurable output transformation pipeline.

Features:

- Format conversion (JSON, YAML, CSV, Markdown)
- Text transformations (case conversion, formatting)
- Code formatting and linting
- Custom transformation scripts
- Validation and schema enforcement
- Output templates

Estimated Complexity: Medium

Time Estimate: 3-4 weeks

Version 5.0 - Collaboration Features (Planned: 2027)

Team Collaboration Hub

Problem: No built-in features for team sharing or collaboration.

Solution: Shared contexts, templates, and workflows for teams.

Features:

- Shared prompt libraries
- Team-wide memory pools
- Collaborative workflow editing
- Usage quota management per team member
- Audit logs and compliance reporting
- Role-based access control

Technical Approach:

- Cloud-based backend for synchronization
- Local-first architecture with sync
- Conflict resolution for concurrent edits
- Encrypted storage for sensitive contexts

Estimated Complexity: Very High

Time Estimate: 12-16 weeks

Community Requests

We're actively collecting feature requests from the community. If you have ideas for new tools or improvements, please:

1. Open an issue on [GitHub Issues](https://github.com/Beaulewis1977/gemini-context-extension/issues) (<https://github.com/Beaulewis1977/gemini-context-extension/issues>)
2. Join discussions on [GitHub Discussions](https://github.com/Beaulewis1977/gemini-context-extension/discussions) (<https://github.com/Beaulewis1977/gemini-context-extension/discussions>)

3. Vote on existing feature requests

Top Requested Features (Community Voting)

Feature	Votes	Status
API key rotation	24	Under consideration
Custom cost alerts (email/ Slack)	19	Under consideration
Browser-based analytics UI	15	Under consideration
Integration with VS Code	12	Researching
Prompt library marketplace	10	Future consideration

Release Timeline

Version	Focus Area	Target Date
1.0	Core tools (Context + Cost)	✓ Released
2.0	Context Optimization	Q1 2026
2.5	Prompt Engineering	Q2 2026
3.0	Multi-Model Intelligence	Q3 2026
3.5	Session Intelligence	Q4 2026
4.0	Workflow Automation	Q1-Q2 2027
5.0	Collaboration	Q3-Q4 2027

Contributing to the Roadmap

Want to influence the roadmap or contribute to development?

- **Prioritization:** Vote on features in GitHub Issues
- **New Ideas:** Open a feature request with the `enhancement` label
- **Development:** Pick an item from this roadmap and submit a PR
- **Sponsorship:** Sponsor features you need urgently

Principles Guiding Development

1. **Minimal Code:** Every feature should be implemented with the fewest lines possible
 2. **Zero Runtime Dependencies:** Use Node.js built-ins whenever possible
 3. **Cross-Platform:** All features must work on WSL, Windows, macOS
 4. **Performance First:** Tools should be fast and non-blocking
 5. **User-Centric:** Solve real problems with simple, intuitive interfaces
-

Last Updated: October 21, 2025

Next Review: January 2026