

- MicroRapid Features
 - Core Features
 - 🚀 Direct API Execution
 - 🔧 Command Line Interface
 - 🎯 Smart Features
 - Intelligent Parameter Mapping
 - Environment Management
 - Request Building
 - Response Handling
 - 📦 Code Generation Features
 - Client SDK Generation
 - Language Support
 - 🔒 Security Features
 - 🛠️ Developer Experience
 - Helpful Output
 - Debugging Support
 - Project Organization
 - 🔄 Integration Features
 - CI/CD Ready
 - Version Control Friendly
 - 📊 Analysis Capabilities
 - 🎨 Customization Options
 - Templates
 - Configuration
 - Advanced Features (Planned)
 - 🤖 AI-Powered Features
 - 📈 Monitoring & Analytics
 - 🔗 Extended Protocol Support
 - 🏢 Enterprise Features
 - Platform Support
 - Operating Systems
 - API Specification Formats
 - File Formats
 - Performance Features
 - Documentation Features

MicroRapid Features

Core Features

Direct API Execution

- Execute OpenAPI 3.0/3.1 operations directly without code
- Execute Swagger 2.0 operations
- Execute GraphQL queries and mutations from schema files
- Execute cURL commands with organization
- No intermediate files or conversion needed

Command Line Interface

- **init** - Initialize new API projects with templates
 - REST API template
 - GraphQL template
 - Download specs from URLs
 - Auto-detect API version from URLs
- **run** - Execute API operations
 - Smart operation detection
 - Multiple data input methods (JSON, files, stdin)
 - Environment-based configuration
 - Request templates with variables
 - Built-in retry and timeout support
 - Dry-run mode with cURL output
- **generate** - Generate production-ready SDKs
 - TypeScript/JavaScript clients
 - Python clients
 - cURL command scripts
 - Postman collections
 - Planned: Go, Rust, Java, C#, Ruby, PHP, Swift, Kotlin

- **analyze** - Analyze API specs
 - Generate request examples
 - Extract operation details
 - Create data templates
- **list** - List API resources
 - Operations from specs
 - Saved request configurations
 - Multiple output formats (table, JSON, YAML)
- **show** - Show operation details
 - Parameter information
 - Request/response schemas
 - Examples generation
 - JSON/YAML output formats
- **explore** - Search API operations
 - Keyword-based search
 - Search across operations, paths, descriptions
 - Grouped results by relevance
- **scaffold** - Generate test infrastructure
 - NPM scripts
 - Makefiles
 - Shell scripts
 - Docker Compose files
 - Direct cURL commands
- **test** - Test API operations
 - Run individual operations
 - Test all operations
 - Automatic cleanup
- **cleanup** - Manage project artifacts
 - Remove test artifacts

- Clean backup directories
- Preserve important files

Smart Features

Intelligent Parameter Mapping

- Automatic path parameter detection
- Query parameter inference
- Common parameter shortcuts (--id, --name, --limit)
- Generic parameter support (--param key=value)

Environment Management

- Separate configs for dev/staging/prod
- Environment-specific base URLs
- Environment variable substitution
- Per-environment authentication

Request Building

- Multiple data input methods
 - Direct JSON: `--data '{"key": "value"}'`
 - File input: `--file data.json` or `--data @data.json`
 - Stdin: `--stdin`
 - Template files with variables
- Required-only mode for quick testing
- Edit mode for interactive data modification

Response Handling

- Multiple output formats (pretty, JSON, YAML, table)
- Save responses to files
- Automatic error formatting
- Status code interpretation



Code Generation Features

Client SDK Generation

- Type-safe methods for all operations
- Built-in error handling
- Authentication support
- Request/response validation
- Auto-generated documentation
- Package configuration files

Language Support

- **Implemented:**

- TypeScript with Fetch API
- Python with Requests
- cURL bash scripts
- Postman collections

- **Planned:**

- Go with net/http
- Rust with reqwest
- Java with OkHttp
- C# with HttpClient
- Ruby with Net::HTTP
- PHP with Guzzle
- Swift with URLSession
- Kotlin with Ktor



Security Features

- API key management
- Bearer token authentication
- Custom header support
- Environment-based secrets
- No hardcoded credentials



Developer Experience

Helpful Output

- Colored terminal output
- Progress indicators
- Clear error messages
- Operation suggestions
- Example commands

Debugging Support

- Verbose mode with request details
- Dry-run to preview requests
- cURL command generation
- Request/response logging

Project Organization

- Standard directory structure
- Request configuration files
- Environment configs
- Data templates
- Test artifacts management



Integration Features

CI/CD Ready

- Exit codes for automation
- JSON output for parsing
- Batch operation support
- Scriptable commands

Version Control Friendly

- Text-based configurations
- Gitignore templates
- No binary artifacts
- Reproducible outputs



Analysis Capabilities

- Operation discovery
- Parameter analysis
- Schema inspection
- Dependency detection
- Usage statistics



Customization Options

Templates

- Request templates with variables
- Custom SDK templates (planned)
- Output format templates
- Project scaffolding templates

Configuration

- Global settings
- Per-project configs
- Environment overrides
- Command aliases (planned)

Advanced Features (Planned)



AI-Powered Features

- Intelligent parameter suggestion
- Error resolution hints
- Optimization recommendations
- Natural language operation search



Monitoring & Analytics

- Request performance tracking
- Error rate monitoring
- Usage analytics
- Cost estimation



Extended Protocol Support

- GraphQL subscriptions
- WebSocket operations
- gRPC services
- AsyncAPI support



Enterprise Features

- Private registry support
- Custom authentication plugins
- Audit logging
- Compliance reporting
- Air-gapped deployment

Platform Support

Operating Systems

- macOS (Intel & Apple Silicon)
- Linux (x64 & ARM)
- Windows (x64)
- Docker containers

API Specification Formats

- OpenAPI 3.0/3.1
- Swagger 2.0
- GraphQL Schema
- AsyncAPI

-  gRPC/Protocol Buffers

File Formats

-  YAML specifications
-  JSON specifications
-  Environment files
-  Template files

Performance Features

- Written in Rust for speed
- Minimal dependencies
- Fast startup time
- Efficient memory usage
- Parallel operation execution (planned)

Documentation Features

- Inline help for all commands
- Example-driven documentation
- API operation discovery
- Auto-generated SDK docs
- Interactive tutorials (planned)