Wisp Roadmap

This document tracks potential improvements, new features, and cleanup opportunities for the Wisp HTTP client library.

Feature Proposals

[Priority: High] Cookie Jar Support

Description: Add automatic cookie management with persistent cookie storage across requests.

Rationale: Currently cookies must be manually set via headers. A proper cookie jar would: - Automatically persist cookies from Set-Cookie headers - Send appropriate cookies based on domain/path - Support cookie persistence to disk for session management - Enable stateful HTTP interactions (login sessions, CSRF tokens)

Affected Files: - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/Core/Request.lean` - Add cookieJar field to Request - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/HTTP/Client.lean` - Integrate cookie jar handling - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/FFI/Easy.lean` - Expose CURLOPT_COOKIEFILE and CURLOPT_COOKIEJAR

Estimated Effort: Medium

Dependencies: None

[Priority: High] WebSocket Support

Description: Add WebSocket client capabilities using libcurl's websocket support (available in curl 7.86+).

Rationale: WebSocket is essential for real-time applications. libcurl now supports WebSocket connections. This would enable: - Bidirectional communication - Real-time updates without polling - Integration with modern APIs that use WebSocket

Affected Files: - New file: `/Users/Shared/Projects/lean-workspace/wisp/Wisp/HTTP/WebSocket.lean` - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/FFI/Easy.lean` - Add websocket FFI bindings - `/Users/Shared/Projects/lean-workspace/wisp/native/src/wisp_ffi.c` - Add curl_ws_* functions

Estimated Effort: Large

Dependencies: libcurl 7.86+ with WebSocket support

[Priority: High] Retry Logic with Exponential Backoff

Description: Add configurable automatic retry for transient failures.

Rationale: Network requests can fail due to transient issues. Automatic retry with exponential backoff would: - Improve reliability for unreliable networks - Handle 429 (Too Many Requests) responses properly - Support configurable retry counts and delays - Allow custom retry conditions (which status codes/errors to retry)

Affected Files: - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/Core/Request.lean` - Add retry configuration - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/HTTP/Client.lean` - Implement retry logic

Estimated Effort: Medium

Dependencies: None

[Priority: Medium] Proxy Support

Description: Add HTTP/HTTPS/SOCKS proxy configuration.

Rationale: Many enterprise environments require proxy usage. libcurl already supports proxies, but the Lean API does not expose this functionality.

Affected Files: - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/Core/Request.lean` - Add proxy configuration to Request - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/HTTP/Client.lean` - Apply proxy settings

Estimated Effort: Small

Dependencies: None (already using CURLOPT_PROXY constant in FFI)

[Priority: Medium] Connection Pooling Configuration

Description: Expose connection pool configuration for better performance tuning.

Rationale: libcurl maintains connection pools internally. Exposing configuration would allow: - Setting max connections per host - Setting total max connections - Configuring keep-alive behavior - Better resource management in high-throughput scenarios

Affected Files: - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/HTTP/Client.lean` - Add pool configuration - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/FFI/Easy.lean` - Add CURLOPT_MAXCONNECTS bindings - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/FFI/Multi.lean` - Add CURLMOPT_* settings

Estimated Effort: Small

Dependencies: None

[Priority: Medium] Progress Callbacks

Description: Add upload/download progress reporting for large file transfers.

Rationale: For large file uploads/downloads, progress feedback is essential for UX. libcurl supports progress callbacks that could be exposed.

Affected Files: - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/FFI/Easy.lean` - Add progress callback FFI - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/HTTP/Client.lean` - Expose progress in API - `/Users/Shared/Projects/lean-workspace/wisp/native/src/wisp_ffi.c` - Implement progress callback wrapper

Estimated Effort: Medium

Dependencies: None

[Priority: Medium] Request/Response Interceptors

Description: Add middleware pattern for request/response transformation.

Rationale: Interceptors enable cross-cutting concerns like: - Automatic request signing - Response logging - Request/response transformation - Metrics collection - Authentication token refresh

Affected Files: - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/HTTP/Client.lean` - Add interceptor chain

Estimated Effort: Medium

Dependencies: None

[Priority: Medium] HTTP/2 and HTTP/3 Configuration

Description: Expose HTTP/2 and HTTP/3 settings for modern protocol support.

Rationale: HTTP/2 and HTTP/3 offer performance benefits. While libcurl may default to these, explicit configuration would allow: - Forcing specific HTTP versions - Enabling ALPN negotiation - HTTP/3 (QUIC) support for low-latency connections

Affected Files: - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/Core/Request.lean` - Already has HttpVersion type but not fully used - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/HTTP/Client.lean` - Apply HTTP version settings - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/FFI/Easy.lean` - Add CURLOPT_HTTP_VERSION binding

Estimated Effort: Small

Dependencies: libcurl built with HTTP/2 (nghttp2) and/or HTTP/3 (ngtcp2/quiche)

[Priority: Medium] DNS-over-HTTPS (DoH) Support

Description: Add configuration for DNS-over-HTTPS resolvers.

Rationale: DoH provides privacy and security for DNS lookups. libcurl supports DoH and this could be exposed.

Affected Files: - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/Core/Request.lean` - Add DoH configuration - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/FFI/Easy.lean` - Add CURLOPT_DOH_URL binding

Estimated Effort: Small

Dependencies: None

[Priority: Low] Request Caching

Description: Add optional response caching with cache-control header respect.

Rationale: Caching can significantly improve performance for cacheable resources. Could integrate with the cellar library from this workspace.

Affected Files: - New file: `/Users/Shared/Projects/lean-workspace/wisp/Wisp/HTTP/Cache.lean` - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/HTTP/Client.lean` - Integrate cache

Estimated Effort: Large

Dependencies: Could optionally use cellar library from workspace

---

[Priority: Low] OAuth 2.0 Helper

Description: Add OAuth 2.0 authentication flow helpers.

Rationale: OAuth 2.0 is ubiquitous for API authentication. Helpers for common flows would simplify integration: - Authorization Code flow - Client Credentials flow - Token refresh handling

Affected Files: - New file: `/Users/Shared/Projects/lean-workspace/wisp/Wisp/Auth/OAuth2.lean`

Estimated Effort: Large

Dependencies: None

[Priority: Low] Request Rate Limiting

Description: Add client-side rate limiting to avoid overwhelming servers.

Rationale: Prevents 429 responses and ensures polite crawling/API usage.

Affected Files: - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/HTTP/Client.lean` - Add rate limiter

Estimated Effort: Medium

Dependencies: None

Code Improvements

[Priority: High] Extract Duplicate Request Setup Logic

Current State: The `execute` and `executeStreaming` functions in `/Users/Shared/Projects/lean-workspace/wisp/Wisp/` contain nearly identical code for setting up curl options (~200 lines duplicated, lines 314-448 and 458-603).

Proposed Change: Extract the common request setup logic into a shared helper function like `setupEasyHandle : Client -> Request -> IO (Wisp.FFI.Easy × Wisp.FFI.Slist)`.

Benefits: - Eliminates code duplication - Makes maintenance easier - Reduces risk of inconsistencies between execution modes - Improves readability

Affected Files: - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/HTTP/Client.lean`

Estimated Effort: Small

[Priority: High] Add Synchronous Execution Option

Current State: The `executeSync` function exists but is not prominently documented, and all convenience methods (`get`, `postJson`, etc.) return `Task`.

Proposed Change: Add synchronous versions of convenience methods (`getSync`, `postJsonSync`, etc.) for simpler use cases where async is not needed.

Benefits: - Simpler API for simple use cases - Reduces boilerplate for scripts and CLI tools - Better developer experience for beginners

Affected Files: - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/HTTP/Client.lean`

Estimated Effort: Small

[Priority: Medium] Improve Error Types with More Context

Current State: `WispError` variants like `curlError` and `ioError` only contain a message string.

Proposed Change: Add structured error data: - Include curl error code as a field (not just in message) - Add request URL to connection errors - Include response body in HTTP errors when available

Benefits: - Enables programmatic error handling - Better debugging experience - Pattern matching on specific errors

Affected Files: - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/Core/Error.lean` - `/Users/Shared/Projects/l`

Estimated Effort: Small

[Priority: Medium] Add JSON Parsing Integration

Current State: Response body is returned as raw `ByteArray` or `String`. JSON parsing must be done externally.

Proposed Change: Add optional JSON parsing helpers: - `Response.bodyJson? : IO (Option Lean.Json)` - Consider integration with a Lean JSON library

Benefits: - Convenient JSON response handling - Type-safe JSON access

Affected Files: - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/Core/Response.lean`

Estimated Effort: Small

Dependencies: Lean.Json or external JSON library

[Priority: Medium] Make Content-Length Calculation More Robust

Current State: In `Client.lean`, content length is set using `content.length` which counts UTF-8 code points, not bytes.

Proposed Change: Use `content.toUTF8.size` for accurate byte length:

```
-- Current (line 360):
Wisp.FFI.setoptLong easy Wisp.FFI.CurlOpt.POSTFIELDSIZE content.length.toInt64
-- Should be:
Wisp.FFI.setoptLong easy Wisp.FFI.CurlOpt.POSTFIELDSIZE content.toUTF8.size.toInt64
```

Benefits: - Correct Content-Length for non-ASCII content - Prevents potential truncation issues

Affected Files: - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/HTTP/Client.lean` (lines 360, 364, 509, 513)

Estimated Effort: Small

[Priority: Medium] Add Request Timeout per Streaming Chunk

Current State: Streaming responses use the overall request timeout. If a server sends data slowly, chunks may arrive indefinitely.

Proposed Change: Add a per-chunk timeout for streaming responses to detect stalled connections.

Benefits: - Better handling of slow/stalled connections - More predictable streaming behavior

Affected Files: - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/Core/Streaming.lean` - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/HTTP/Client.lean`

Estimated Effort: Medium

[Priority: Low] Add Request/Response Debug Logging

Current State: Verbose mode enables curl's debug output to stderr, but there's no structured logging.

Proposed Change: Add optional structured logging that can be directed to a custom logger: - Request method, URL, headers - Response status, timing, headers - Error details

Benefits: - Better debugging support - Integration with application logging

Affected Files: - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/HTTP/Client.lean`

Estimated Effort: Medium

[Priority: Low] Lazy Global Initialization

Current State: `globalInit` must be called explicitly before using the library. While `easyInit` calls it automatically, this is undocumented behavior.

Proposed Change: Document the automatic initialization clearly and consider using `IO.initializing` pattern for truly lazy init.

Benefits: - Better developer experience - Less boilerplate in examples

Affected Files: - `/Users/Shared/Projects/lean-workspace/wisp/README.md` - `/Users/Shared/Projects/lean-workspa`

Estimated Effort: Small

Code Cleanup

[Priority: High] Remove Unused HttpVersion Type

Issue: The `HttpVersion` type is defined in `/Users/Shared/Projects/lean-workspace/wisp/Wisp/Core/Types.lean` (lines 38-44) but is never used anywhere in the codebase.

Location: `/Users/Shared/Projects/lean-workspace/wisp/Wisp/Core/Types.lean:38-44`

Action Required: Either: 1. Remove the unused type, or 2. Implement HTTP version configuration using this type (preferred)

Estimated Effort: Small

[Priority: Medium] Add Missing Status Codes to statusText

Issue: The `statusText` function in `/Users/Shared/Projects/lean-workspace/wisp/Wisp/Core/Response.lean` (lines 75-103) is missing several common HTTP status codes.

Location: `/Users/Shared/Projects/lean-workspace/wisp/Wisp/Core/Response.lean:75-103`

Action Required: Add missing status codes: - 206 Partial Content - 301 Moved Permanently (missing descriptive handling) - 405 Method Not Allowed - 406 Not Acceptable - 411 Length Required - 413 Payload Too Large - 415 Unsupported Media Type - 422 Unprocessable Entity - 451 Unavailable For Legal Reasons

Estimated Effort: Small

[Priority: Medium] Consolidate Example Files

Issue: The examples directory has four files (`SimpleGet.lean`, `PostJSON.lean`, `ClientTest.lean`, `MinimalTest.lean`) with overlapping functionality and inconsistent styles.

Location: `/Users/Shared/Projects/lean-workspace/wisp/examples/`

Action Required: 1. Keep `SimpleGet.lean` as the primary example 2. Merge `PostJSON.lean` functionality into a comprehensive `Examples.lean` 3. Move `MinimalTest.lean` and `ClientTest.lean` to a tests/ subdirectory or remove (redundant with test suite)

Estimated Effort: Small

[Priority: Medium] Add Deriving Clauses for Better Debugging

Issue: Several types lack `Repr` derivations making debugging harder.

Location: - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/Core/Request.lean` - `MultipartPart`, `Body`, `Auth`, `SslOptions` - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/Core/Response.lean` - `Response` - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/Core/Streaming.lean` - `StreamingResponse`

Action Required: Add `deriving Repr` where missing, or implement `ToString` instances.

Estimated Effort: Small

[Priority: Low] Improve FFI Header Documentation

Issue: The C header file (`wisp_ffi.h`) has minimal documentation for function parameters and return values.

Location: `/Users/Shared/Projects/lean-workspace/wisp/native/include/wisp_ffi.h`

Action Required: Add Doxygen-style comments documenting: - Parameter purposes - Return value semantics - Error conditions - Thread safety considerations

Estimated Effort: Small

[Priority: Low] Add BEq Instance for WispError

Issue: `WispError` in `/Users/Shared/Projects/lean-workspace/wisp/Wisp/Core/Error.lean` lacks `BEq` and `Hashable` instances.

Location: `/Users/Shared/Projects/lean-workspace/wisp/Wisp/Core/Error.lean:174-182`

Action Required: Add `deriving BEq` to WispError or implement manual instance.

Estimated Effort: Small

[Priority: Low] Update Lean Version Requirement in README

Issue: README states “Lean 4.25.0 or later” but `lean-toolchain` uses 4.26.0.

Location: `/Users/Shared/Projects/lean-workspace/wisp/README.md:29`

Action Required: Update README to match actual lean-toolchain version.

Estimated Effort: Small

Testing Improvements

[Priority: Medium] Add Unit Tests for Edge Cases

Issue: Test suite focuses on integration tests against httpbin.org. Missing unit tests for: - Header parsing edge cases (malformed headers, unusual encodings) - SSE parser edge cases (partial chunks, missing fields) - URL encoding edge cases - Error handling paths

Location: `/Users/Shared/Projects/lean-workspace/wisp/Tests/Main.lean`

Action Required: Add unit tests that don’t require network access.

Estimated Effort: Medium

[Priority: Medium] Add Timeout Tests with Mock Server

Issue: Timeout tests depend on httpbin.org’s `/delay` endpoint which can be unreliable.

Location: `/Users/Shared/Projects/lean-workspace/wisp/Tests/Main.lean:232-254`

Action Required: Consider using a local mock server for timeout testing reliability.

Estimated Effort: Medium

[Priority: Low] Add Performance Benchmarks

Issue: No performance benchmarks exist to track throughput or latency.

Action Required: Add benchmarks for: - Sequential request throughput - Concurrent request throughput - Large file download speed - Memory usage under load

Estimated Effort: Medium

Documentation Improvements

[Priority: Medium] Add API Documentation Comments

Issue: Most public functions lack doc comments explaining parameters and behavior.

Location: Throughout the codebase, especially: - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/HTTP/Client.le` - `/Users/Shared/Projects/lean-workspace/wisp/Wisp/Core/Request.lean`

Action Required: Add docstrings to all public functions and types.

Estimated Effort: Medium

[Priority: Low] Add CONTRIBUTING.md

Issue: No contribution guidelines exist.

Action Required: Create `/Users/Shared/Projects/lean-workspace/wisp/CONTRIBUTING.md` with: - Development setup instructions - Testing guidelines - Code style guidelines - PR process

Estimated Effort: Small

[Priority: Low] Add CHANGELOG.md

Issue: No changelog tracking version history.

Action Required: Create `/Users/Shared/Projects/lean-workspace/wisp/CHANGELOG.md` following Keep a Changelog format.

Estimated Effort: Small