**PicoFramework Feature Checklist**

This document provides an overview of implemented features, utilities, and system components in the PicoFramework.

**Core Architecture**

* Modular MVC-inspired structure
* FrameworkApp base class for application entry
* FrameworkController base class for route logic
* FrameworkTask for FreeRTOS task abstraction
* FrameworkManager for centralized initialization

**HTTP Server**

* Lightweight embedded HTTP server
* Routing system with method/path matching
* Middleware support (per-route, chainable)
* Request & Response abstractions
* Static file serving (SD/FatFs)
* Multipart form parsing
* MIME type detection

**HTTP Client**

* HttpRequest with fluent builder API
* Full URL parsing with .setUri()
* TLS support with root certificate verification
* HttpResponse object with status, headers, body, file-saving
* Chunked transfer decoding
* HttpClient abstraction under-the-hood

**Application Support**

* AppContext for global service access
* Configuration of network, time, storage
* JsonService for JSON file persistence
* FrameworkModel for CRUD-style persistence
* FrameworkView for basic templating/HTML rendering

**Event System**

* Event struct with type, payload, and source
* EventManager with pub/sub model
* FreeRTOS task notification integration
* onEvent() handler for task-based event delivery

**Timer Service**

* One-shot timestamp-based scheduling
* Recurring interval scheduling
* Daily scheduling with time + day mask
* Start/stop event scheduling with duration
* Job cancellation by ID
* Automatic job ID tracking
* Missed job retry after reboot *(planned)*
* Persistent job store *(planned)*

**Time and RTC Support**

* TimeManager with NTP and DS3231 RTC sync
* PicoTime utility class for formatting/conversion
* NTPClient with DNS, retry, and UTC offset
* Works on both RP2040 and RP2040+RTC boards

**Storage Support**

* Abstract StorageManager interface
* FatFsStorageManager implementation for SD card
* File operations: read, write, append, mkdir, exists, remove

**Logging**

* Logger with:
  + Console or SD output
  + Log levels: INFO, WARN, ERROR
  + Timestamps on entries
* Planned: Log rotation support

**Debug Tracing**

* Macro-based tracing system
* Per-module trace enablement
* Trace level filtering (build-time or runtime)
* Optional timestamps
* Output to console or SD
* Configured via framework\_config.h

**Utilities**

* URL parsing, decoding helpers
* MIME type detection
* TCP memory + state diagnostics
* Runtime FreeRTOS task stats
* Heap/stack reporting
* cppMemory allocation tracking
* Idle memory measurement tool

**Testing (In Progress)**

* CppUTest framework integration
* Unit tests for core components (router, controller, etc.)
* End-to-end HTTP route tests
* Memory-safe testing of chunked/TLS parsing

**Documentation**

* Full Doxygen comments for:
  + Public headers
  + Core framework classes
  + Utility functions
  + File-level doc blocks (with license/author)
* Auto-generated HTML/PDF *(planned)*

**Build and Environment**

* CMake-based modular build system
* Clean source structure for framework and app
* Ready for Raspberry Pi Pico W (FreeRTOS + lwIP)

**Example App (In Progress)**

* Login endpoint
* Token-based JWT authentication
* HTML frontend from SD
* GPIO control through web UI

**JWT Authentication**

* Middleware-based token checking
* Authorization: Bearer <jwt> parsing
* HMAC signature validation
* Token expiry verification
* Protected route support with fallback 401 response
* JWT secret from build or config

**Routing System (Express.js-style)**

* Add routes via router.addRoute(method, path, handler)
* REST-friendly: GET, POST, PUT, DELETE
* Lambdas or method pointers as handlers
* Per-route and global middleware
* Path parsing and matching with variables
* Built-in /auth route for token test (can override)
* Route debug printing

**Example**

router.addRoute("GET", "/info", [](Request &req, Response &res) {

res.sendText("System Info Page");

});

router.addRoute("POST", "/config", configHandler, {authMiddleware});

This feature set defines PicoFramework v1 and is suitable for secure, modular, embedded web APIs and local control interfaces.