**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

**Application Support**

* AppContext for global service access
* Configuration of network, time, storage
* JsonService for JSON file persistence
* FrameworkModel for CRUD persistence
* FrameworkView (light templating / HTML serving)

**Event System**

* Event struct with type, payload, and source
* EventManager with publish/subscribe model
* Event delivery using FreeRTOS task notifications
* Task onEvent() support for receiving events

**TimerService**

* Schedule one-shot events by timestamp
* Schedule recurring events by interval
* Schedule daily events by time and day mask
* Schedule start/stop events with duration
* Cancel scheduled jobs by job ID
* Built-in job ID management
* Will retry missed jobs after reboot (planned)
* Persistence of scheduled jobs (planned for future)

**Time and RTC Support**

* TimeManager with NTP + RTC (DS3231)
* PicoTime utility class for conversion/formatting
* NTPClient with retry and DNS support
* Compatible with both RP2040 and RP2040+RTC

**Storage Support**

* StorageManager interface
* FatFsStorageManager implementation
* File read, write, append, mkdir, exists, remove

**Logging**

* Logger class with:
  + Console or SD card output
  + Log levels (INFO, WARN, ERROR)
  + Timestamped logs
* Log rotation (future)

**Debug Tracing**

* Lightweight macro-based tracing system
* Per-module trace enablement
* Trace level filtering
* Optional timestamp in output
* Output to SD or console
* Configured via framework\_config.h

**Utilities**

* URL parsing, decoding
* MIME detection
* TCP state & memory diagnostics
* Runtime task stats
* Heap info and PCB display
* cppMemory allocator tracking
* Idle memory measurement

**Testing (In Progress)**

* CppUTest integration
* Unit tests for Router, Controller, Request, etc.
* End-to-end route + HTTP tests

**Documentation**

* Full Doxygen comments for:
  + All public headers
  + All core classes
  + Utility functions
  + File-level doc blocks with author/license
* Auto-generated HTML/PDF docs (planned)

**Build / Environment**

* CMake-based build system
* Modular file structure
* Ready for Raspberry Pi Pico W

**Example App (In Progress)**

* Login endpoint
* Token-based auth using JWT
* HTML frontend served from SD
* GPIO control via Web UI

**JWT Authentication**

* Built-in support for JWT-based route protection
* Authorization header parsing with "Bearer" token format
* JWT token decoding and signature validation
* Middleware integration for protected routes
* Optional token expiry checking

Tokens are validated using a secret key, typically defined at build time or loaded from config. Middleware automatically checks authorization and responds with 401 if token is invalid or missing.

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

* Add routes using router.addRoute(method, path, handler)
* Full REST API enabled
* Supports GET, POST, PUT, DELETE, etc.
* Lambdas or bound methods as route handlers
* Per-route and global middleware support
* URL path normalization and matching
* Helper for extracting path/query/form/cookie data
* handleRequest() dispatches requests to matched handler
* Built-in /auth route pattern (optional override) for JWT token testing
* Route printing for debug output

**Example**

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

res.sendText("System Info Page");

});

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

Routes are matched by method and path. Supports path arguments. Matching is fast and embedded-safe.