# Working CLI Documentation

**Written: 2023 - 10 - 29**

## 1. Config.js

config.js is a crucial module used for managing configuration settings for the CLI application.

Configuration Data: This file stores key data such as the name of the CLI application, its version, description, and other configuration options like the superuser or database information.

Settings Retrieval: Other parts of the application can import config.js to access these configuration settings when needed.

## 2. Express.js

Web Server Setup: This module involves the configuration and initialization of an Express.js server, which includes defining routes, middleware, and other server-related settings.

API Endpoints: It defines routes and endpoints for the web server, enabling the CLI application to provide a web-based interface for specific tasks or data retrieval.

HTTP Request Handling: The script manages incoming HTTP requests and responds with appropriate data or actions.

## 3. Init.js

init.js is a module responsible for initializing the CLI project.

Folder Structure Creation: Creating the necessary project folder structure based on your project's requirements.

Configuration File Creation: This module generates and populates configuration files, including config.json, with default settings.

## 4. logEvents.js

logEvents.js is a module responsible for managing and recording log events within the CLI application.

Logging Framework: Implementing a logging framework to capture and store log events.

Event Handling: Setting up event listeners to manage log events, including their level (e.g., INFO, WARNING, ERROR).

Storage or Output: Determining where and how log events are stored or displayed, which could include options like console output, log files, or a remote log service.

## 5. Myapp.js

myapp.js is the primary entry point for the CLI application, responsible for handling command-line interactions.

Command Line Argument Processing: myapp.js captures and processes command-line arguments and options provided by users when running the CLI.

Command Routing: This file routes user commands to relevant functions or modules based on the arguments provided.

Error Handling: It takes care of error handling, displaying user-friendly error messages when users enter incorrect commands or formats.

Integration: This script interacts with other modules and files, such as config.js, express.js, and init.js, depending on the specific user commands.

Help Command: Provides a help command (myapp --help), offering instructions and displaying the available commands to guide users.

**6. Server.js**

- Server Setup: Initializes an HTTP server using the 'http' module.

- Routing Logic: Handles various request types using a switch statement.

- File Handling: Serves HTML files using the 'fs.readFile' method.

- POST Request Processing: Manages data from POST requests for the '/new' endpoint.

- Token Count Endpoint: Retrieves the current token count for '/count'.

- Response Generation: Generates and sends HTML responses to clients.

- Error Handling: Provides basic error handling for file reading errors.

**7. Templates.js**

templates.js contains various template strings that are likely used for displaying information and instructions to users of the CLI application.

Usage Instructions: This has the usage instructions for different commands and options in the CLI.

**8. Token.js**

token.js is a module related to managing tokens in your CLI application. It includes functions for token creation, updating, counting, and searching.

Token Creation: Functions to generate new tokens with specific attributes, such as creation date, username, email, phone, and expiration.

Token Counting: A function to count and display the number of tokens currently in use.

Token Updating: Functions to update token information, such as phone or email.

Token Listing and Fetching: Functions to list all tokens and fetch a token for a specific username.

Token Searching: Functions to search for tokens based on different criteria like username, email, or phone number.