Webmon Implementation

1. Introduction :

This report details the implementation of the webmon component, a monitoring system for the Unreliable Banking Server (UBS). The implementation follows the requirements specified in Assignment #2, creating a robust monitoring solution that manages and monitors the UBS server

2. System Architecture

2.1. Component Overview:

The system consists of three main components:

- **Webmon (Monitor Component)**
- **UBS (Unreliable Banking Server)**
- **Configuration System (webmon.json)**

2.2. File Strutcute:

```
assignment-2/
├── webmon.py         # Main monitoring component
├── server.py         # UBS server from Assignment 1
├── webmon.json       # Configuration file
├── requirements.txt # Dependencies
├── webmon.log       # Webmon event logs
└── server.log       # UBS server logs
```

3. Implementation Details:

3.1.

```
def __init__(self, config_file="webmon.json"):
    self.config = self._load_config(config_file)
    self.ubs_process = None
    self.setup_logging()
```

Purpose:

1. Initializes the monitoring system
2. Loads configuration
3. Sets up logging infrastructure

3.2. Configuration Loading:

```
def _load_config(self, config_file):
    """Load and validate the configuration file"""
    try:
        with open(config_file, 'r') as f:
```

Purpose:

- Reads and validates webmon.json
- Ensures all required configuration fields are present
- Validates configuration format
- Provides meaningful error messages for invalid configurations

3.3. Then we have : For Server management

```
def start_ubs(self)
def stop_ubs(self)
def restart_ubs(self)
```

Purpose:

- Manages UBS process lifecycle
- Ensures clean process management
- Handles process termination and startup
- Logs all server state changes

3.4. For Monitoring Logic:

```
def check_server_status(self)
def handle_response(self)
```

Purpose:

- Implements core monitoring functionality
- Handles different server responses
- Applies configuration rules
- Manages timeout scenarios
- Implements retry logic

4. Configuration System (webmon.json):

- **waittime: Defines timeout threshold in milliseconds**
- **logto: Specifies logging destination**
- **Response configurations:**
  - **retrytimes: Number of retries before action**
  - **action: Response action (restart/nothing)**

5. **Logging System:**
   5.1. The system implements two separate log files:

Webmon.log -

webmon.log

```
2024-12-10 15:46:20,399 - Started UBS server
2024-12-10 15:46:24,459 - Restarting server after 2 occurrences of http403
2024-12-10 15:46:24,463 - Stopped UBS server
2024-12-10 15:46:25,470 - Started UBS server
2024-12-10 15:46:27,473 - Restarted UBS server
2024-12-10 15:46:47,772 - Timeout exceeded waittime - immediate restart required
2024-12-10 15:46:47,780 - Stopped UBS server
```

Purpose:

- Records monitor actions
- Tracks server state changes
- Documents restart events
- Timestamps all monitor activities

Server.log -

server Close (⌘W)

```
2024-12-10 15:46:22,442 - 127.0.0.1 - 200 OK
2024-12-10 15:46:23,450 - 127.0.0.1 - 403 Forbidden
2024-12-10 15:46:24,458 - 127.0.0.1 - 403 Forbidden
2024-12-10 15:46:28,485 - 127.0.0.1 - 403 Forbidden
2024-12-10 15:46:29,495 - 127.0.0.1 - 200 OK
2024-12-10 15:46:30,499 - 127.0.0.1 - 500 Internal Server Error
```

Purpose:

- Records UBS responses

- Tracks all server interactions
- Documents HTTP status codes
- Maintains client request history

Requirements Fulfillment:

✓ Directory Structure
- Implementation placed in assignment-2 directory
- All components properly organized

✓ Process Management
- Webmon runs in separate process from UBS
- No HTTP listener in webmon
- Clean process separation maintained

✓ Configuration Management
- Reads webmon.json at startup
- Validates configuration
- Applies configuration rules

4.2 Monitoring Requirements

✓ Server Monitoring
- Continuous monitoring loop
- Response checking
- Status verification
- Proper error handling

✓ Response Handling
- Immediate restart on timeout
- Retry logic for HTTP errors
- Proper action implementation

- Clean restart procedures

4.3 Logging Requirements

✓ File System Logging

- Implemented as required
- Proper timestamp formatting
- Comprehensive event logging
- Clear log separation

Also all codes are fully commented in all files ☺