WaveAlert360 Detailed Components

Top data sources / files:
- NOAA Tides & Currents API — tidesandcurrents.noaa.gov
- GitHub Repository — prasannapad/wavealer360
- Mock NWS API — /api/alerts/active
- Test Scenarios — coastal_flood_warning, high_surf_advisory, rip_current_statement
- File: /tmp/led_service.log

Auto-Updater (right column):
- Current vs Latest Commit?
  - Different → Stop Web Dashboard → Create Backup .tar in backup/ → git pull origin main → Pull Successful?
    - Yes → Restart Web Dashboard → Health Check Web Dashboard → No Update Needed
    - No → Log Update Failure Continue with Old Code → Restart Web Dashboard
  - Same → No Update Needed

National Weather Service API — alerts.weather.gov
- Return Alert Level SAFE/CAUTION/DANGER

Parse Alert Response (left column):
- Determine Alert Level SAFE/CAUTION/DANGER
- Write to /tmp/led_control_signal PATTERN:color
- Play Audio via mpg123 normal/caution/high_alert.mp3
  - Log Check Status Timestamp Level Location
  - Audio Output mpg123 player
- Sleep until next check

Watchdog Start:
- Acquire Process Lock?
  - Success → Start All Processes: LED Service, Main Process, Web Dashboard, Auto-Updater
  - Failed → Exit - Another Instance Running
  - Creates → File: .watchdog.lock

Monitor Loop Every 60 seconds:
- Check Each Process → Is Process Running?
  - No → Restart Count less than 5?
    - Yes → Restart Process → Increment Restart Count
    - No → Wait for Cooldown
  - Check Timer 15 seconds → 15s elapsed → Check Audio Updates manifest.json
    - Azure Blob Storage Audio Files
    - Download Updated Audio Files if needed
    - Log Watchdog Status
  - Yes → HTTP GET to Device Service API
    - Device Service API /api/alert/mac

LED Service Start:
- Acquire Process Lock?
  - Success → Initialize Hardware?
    - Try → Import rpi_ws281x
      - Success → Initialize 3 LED Strips: Red GPIO 18, Yellow GPIO 21, Green GPIO 13
        - Success → hardware_available = true
      - Error → hardware_available = false Simulation Mode
  - Failed → Exit - Another Instance Running
  - Creates → File: /tmp/led_service.lock

Monitor Loop Every 2 seconds:
- Read /tmp/led_control_signal
  - File: /tmp/led_control_signal
- Parse Signal Type?

System Startup:
- Load Configuration settings.json devices.json
- Get Device MAC Address from wlan0

Web Dashboard Start:
- Initialize Flask App Port 5000
- Load System Data: LED Status, Device Config, Git Info
  - Config Files: devices.json settings.json
- HTTP Routes
  - GET / HTML Dashboard
  - GET /status JSON API
  - GET /health Health Check
- Verify IP Allowed?
  - Allowed → Serve Response
  - Denied → HTTP 403 Forbidden

Auto-Updater Start:
- Acquire Process Lock?
  - Success → Update Loop Every 120 seconds → Fetch Latest Commit from GitHub API
  - Failed → Exit - Another Instance Running
  - Creates → File: .updater.lock