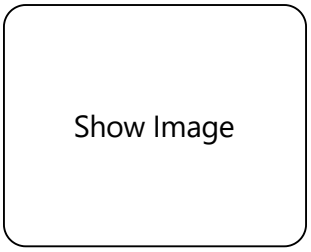
🌺 Giant Sloth Orchard Weather Station v2.0

A complete, professional weather monitoring system with support for WeatherLink v2 Cloud API, Local API, and realistic demo mode. Built for the Giant Sloth Orchard in Hawaii with a beautiful tropical-themed interface.



✨ Features

🌊 Multi-Mode Data Sources

- **Demo Mode**: Realistic weather data simulation with daily cycles and events
- **Cloud API**: WeatherLink v2 API with proper authentication and signatures
- **Local API**: Direct connection to WeatherLink Live devices on your network
- **Auto Mode**: Intelligent fallback (Cloud → Local → Demo)

📊 Advanced Dashboard

- **Real-time Widgets**: Temperature, humidity, pressure, wind, UV, rainfall, solar radiation, battery
- **Visual Gauges**: Circular temperature/humidity/UV gauges with smooth animations
- **Wind Compass**: Live wind direction with cardinal directions
- **Battery Indicator**: Visual battery level with animated fill
- **Interactive Charts**: Historical data visualization with multiple time ranges

⚙️ Configurable Settings

- **Update Intervals**: Live updates (1 second minimum), Historical saves (15 minutes minimum)
- **Widget Management**: Enable/disable individual widgets
- **API Configuration**: Cloud and local API settings with connection testing
- **Data Management**: Export JSON/CSV, import data, clear history

🔧 Technical Features

- **CORS Proxy Server**: Handles WeatherLink API authentication and local device communication
- **Data Persistence**: Automatic historical data storage with 30-day retention
- **Responsive Design**: Works on desktop and mobile devices

- **Performance Monitoring**: API call statistics, success rates, response times
- **Network Discovery**: Automatic detection of WeatherLink devices on your network

🚀 Quick Start

Prerequisites

- Node.js 14.0.0 or higher
- npm 6.0.0 or higher
- WeatherLink device (for local mode) or WeatherLink.com account (for cloud mode)

Installation

1. **Clone or download the files**:

   bash

   ```
   # If using git
   git clone https://github.com/giant-sloth-orchard/weather-station.git
   cd weather-station

   # Or extract the downloaded files to a folder
   ```

2. **Install dependencies**:

   bash

   ```
   npm install
   ```

3. **Create public folder and add the HTML file**:

   bash

   ```
   mkdir public
   # Copy the weather station HTML file to public/index.html
   ```

4. **Start the server**:

   bash

   ```
   npm start
   ```

5. **Open your browser**:
   - Navigate to `http://localhost:3000`
   - The weather station will start in Demo Mode

🔧 Configuration

Demo Mode (Default)

Demo mode works immediately with realistic simulated data:

- Daily temperature cycles based on time of day
- Weather events (rain, clouds, wind variations)
- Optional extreme weather simulation
- No configuration required

Cloud API Mode

For WeatherLink.com integration:

1. **Get API Credentials**:
   - Sign up at WeatherLink.com
   - Go to Account → API Keys
   - Generate API Key and API Secret
   - Note your Station ID
2. **Configure in Settings**:
   - Click the gear icon (⚙) in the top-right
   - Enter your API Key, Station ID, and API Secret
   - Click "Test Connection" to verify
   - Switch to "Cloud API" mode

Local API Mode

For direct device connection:

1. **Find Your Device**:
   - Use the Network Discovery feature in Settings
   - Or check your router's admin panel for WeatherLink devices
   - Note the IP address
2. **Configure in Settings**:
   - Enter the device IP address (e.g., 192.168.1.100)
   - Set the port (usually 80)
   - Enable HTTPS if your device supports it
   - Click "Test Connection" to verify
   - Switch to "Local API" mode

📊 Data Management

Historical Data

- Automatically saved every 15 minutes (configurable)
- Stored in browser localStorage
- 30-day automatic retention
- Export to JSON or CSV formats

Update Intervals

- **Live Updates**: 1 second to 5 minutes (affects widget refresh rate)
- **History Saves**: 15 minutes to 6 hours (affects data storage frequency)

Export/Import

- **Export JSON**: Complete data with settings and history
- **Export CSV**: Historical data in spreadsheet format
- **Import**: Restore previously exported data

🌐 API Endpoints

The proxy server provides several useful endpoints:

Data Endpoints

- `GET /api/demo/current` - Generate demo weather data
- `GET /api/cloud/*` - Proxy to WeatherLink v2 Cloud API
- `GET /api/weather/*` - Proxy to local WeatherLink device

Management Endpoints

- `GET /api/status` - Server status and configuration
- `GET /api/health` - Health check
- `GET /api/discover` - Network device discovery
- `POST /api/test/cloud` - Test cloud API connection
- `POST /api/test/local` - Test local API connection

🔧 Advanced Setup

Production Deployment

1. **Using PM2 (Recommended)**:

bash

```
npm install -g pm2
pm2 start proxy-server.js --name "weather-station"
pm2 startup
pm2 save
```

2. **Using systemd (Linux)**:

bash

```
# Create service file
sudo nano /etc/systemd/system/weather-station.service

# Add configuration (see example in docs)
sudo systemctl enable weather-station
sudo systemctl start weather-station
```

3. **Environment Variables**:

bash

```
export PORT=3000
export NODE_ENV=production
export WEATHERLINK_API_KEY=your_key
export WEATHERLINK_API_SECRET=your_secret
```

Reverse Proxy Setup (Nginx)

nginx

```
server {
    listen 80;
    server_name weather.yourdomain.com;

    location / {
        proxy_pass http://localhost:3000;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}
```

🧪 Testing & Development

Run Tests

```bash
npm test                    # Test all APIs
npm run test:cloud          # Test cloud API only
npm run test:local          # Test local API only
```

Development Mode

```bash
npm run dev                 # Start with auto-reload
```

Health Checks

```bash
npm run health              # Check server health
npm run status              # Get detailed status
npm run demo                # Test demo data generation
```

Discovery

```bash
npm run discover            # Find WeatherLink devices on network
```

📱 Mobile Usage

The weather station is fully responsive and works great on mobile devices:

- Touch-friendly interface
- Optimized layouts for small screens
- Swipe gestures supported
- Fast loading and smooth animations

🛠️ Troubleshooting

Common Issues

1. **"Cloud API Error"**:
   - Verify API credentials are correct

- Check WeatherLink.com account status
- Ensure station is online and transmitting

2. **"Local API Error"**:
   - Verify device IP address
   - Check network connectivity
   - Ensure device is powered on
   - Try network discovery feature
3. **"No historical data"**:
   - Wait for automatic data collection (15+ minutes)
   - Check browser localStorage isn't full
   - Verify data save interval settings
4. **Charts not loading**:
   - Ensure JavaScript is enabled
   - Check browser console for errors
   - Try refreshing the page

Performance Optimization

1. **Reduce update frequency** for battery-powered devices
2. **Clear old data** regularly if storage is limited
3. **Use local mode** when possible for faster response
4. **Monitor performance stats** in settings panel

🔒 Security Notes

- API keys are transmitted securely through the proxy server
- Local network traffic should be on trusted networks only
- Consider HTTPS for production deployments
- Regularly rotate API keys for cloud services

📝 Data Privacy

- All data is stored locally in your browser
- No data is transmitted to third parties (except WeatherLink.com for cloud mode)
- Export your data regularly for backups
- Clear data feature permanently removes all stored information

🤝 Contributing

We welcome contributions! Please:

1. Fork the repository
2. Create a feature branch
3. Make your changes
4. Add tests if applicable
5. Submit a pull request

📄 License

MIT License - see LICENSE file for details.

🆘 Support

- **Issues**: GitHub Issues
- **Email**: tech@giantslothorchard.com
- **Documentation**: This README and inline code comments

🏆 Acknowledgments

- Davis Instruments for WeatherLink API
- Chart.js for beautiful visualizations
- Unsplash for tropical background imagery
- The open-source community for inspiration

🌺 Built with aloha in Hawaii for the Giant Sloth Orchard 🦥

Professional weather monitoring for tropical agriculture