🚀 Quick Start Guide - Rhine Water Level Monitor

Getting Started in 3 Steps

1️⃣ Install Dependencies

```
npm install
```

2️⃣ Start the Server

```
npm start
```

3️⃣ Open in Browser

Navigate to: **http://localhost:3000**

🎯 What You'll See

Current Water Level Display

- **Large number** showing current water level in centimeters
- **Color-coded status badge**:
  - 🟢 **Normal** (< 400 cm) - Green
  - 🟠 **Warnung** (400-800 cm) - Orange
  - 🔴 **Gefahr** (> 800 cm) - Red

24-Hour History Chart

- Interactive line chart showing water level trends
- Threshold lines at 400 cm (warning) and 800 cm (danger)
- Hover over chart for detailed values

Controls

- **Aktualisieren** button - Manual refresh
- **Auto-Aktualisierung** toggle - Enable/disable auto-refresh (60s interval)

📊 Current Rhine Water Level

The app displays **real-time** data from:

```
https://www.stadt-koeln.de/interne-dienste/hochwasser/pegel_ws.php
```

🔧 Troubleshooting

Server won't start?

```
# Check if port 3000 is already in use
# Kill any process using port 3000, then:
npm start
```

No data showing?

1. Check your internet connection
2. Ensure the Cologne API is accessible
3. Check browser console for errors (F12)

Chart not displaying?

1. Ensure Chart.js loaded (check browser console)
2. Try hard refresh (Ctrl+Shift+R)
3. Clear browser cache

💾 Data Storage

The app automatically stores the last 24 hours of readings in your browser's localStorage. This allows:

- Viewing historical data even offline
- Faster chart rendering
- Data persistence across sessions

Clear stored data:

Open browser console (F12) and run:

```
localStorage.removeItem('rhein-pegel-history');
```

⌨️ Keyboard Shortcuts

- **R** - Manual refresh

📱 Mobile Support

The app is fully responsive and works on:

- 📱 Mobile phones
- 📱 Tablets
- 💻 Desktop computers

🌐 Deployment Options

Option 1: GitHub Pages (Static - requires CORS proxy)

```
# Push to GitHub
git init
git add .
git commit -m "Initial commit"
git push origin main

# Enable GitHub Pages in repository settings
```

Option 2: Heroku (with Node.js server)

```
# Install Heroku CLI, then:
heroku create your-app-name
git push heroku main
```

Option 3: Netlify

```
# Install Netlify CLI
npm install -g netlify-cli

# Deploy
netlify deploy --prod
```

🎨 Customization

Change Alert Thresholds

Edit `js/app.js` lines 8-30:

```
const ALERT_LEVELS = {
  NORMAL: { max: 400, ... },    // Change 400 to your value
  WARNING: { max: 800, ... },   // Change 800 to your value
  // ...
};
```

Change Refresh Interval

Edit `js/app.js` line 38:

```
refreshInterval: 60000,  // Change to desired milliseconds
```

Change Colors

Edit `css/main.css` lines 6-12:

```
--color-normal: #4CAF50;   /* Green */
--color-warning: #FF9800;  /* Orange */
--color-danger: #F44336;   /* Red */
```

📖 Documentation

- **Architecture**: See ARCHITECTURE.md
- **Technical Specs**: See TECHNICAL_SPEC.md
- **Full README**: See README.md

☑ Feature Checklist

- ✅ Real-time water level display
- ✅ 24-hour historical chart
- ✅ Color-coded alert system
- ✅ Auto-refresh (60s)
- ✅ Manual refresh button
- ✅ German language interface
- ✅ Responsive design
- ✅ localStorage persistence
- ✅ CORS proxy server
- ✅ Error handling
- ✅ Loading indicators

🆘 Support

For issues or questions:

1. Check the README.md troubleshooting section

2. Review browser console errors (F12)
3. Check server terminal output
4. Verify API accessibility

📊 Current Status

The app is fully functional and tested with:

- ✅ Live data from Cologne API
- ✅ Real-time updates
- ✅ Chart visualization
- ✅ All alert levels
- ✅ Responsive design
- ✅ Auto-refresh mechanism

Enjoy monitoring the Rhine water levels! 🌊