# WhatsApp Automation Web App - GitHub Setup Guide

## Step 1: Create a GitHub Repository

1. Go to GitHub and sign in or create an account if needed
2. Click the "+" button in the top right corner and select "New repository"
3. Name your repository (e.g., "whatsapp-automation-app")
4. Add a description: "Web application for automating WhatsApp message sending"
5. Choose "Public" or "Private" visibility as you prefer
6. Check "Add a README file"
7. Click "Create repository"

## Step 2: Set Up Your Local Environment

1. Install Git on your computer if you don't already have it
2. Open a terminal or command prompt
3. Create a new folder for your project:

```
mkdir whatsapp-automation-app
cd whatsapp-automation-app
```

4. Initialize Git:

```
git init
```

5. Connect to your GitHub repository (replace YOUR_USERNAME with your GitHub username):

```
git remote add origin https://github.com/YOUR_USERNAME/whatsapp-automation-app.git
```

## Step 3: Create Project Files

1. Create the app.py file:

```
touch app.py
```

2. Create a requirements.txt file:

```
touch requirements.txt
```

3. Create a directory for templates:

```
mkdir templates
touch templates/index.html
```

4. Create a README.md file:

```
touch README.md
```

## Step 4: Add Code to Project Files

### In app.py:

python

```python
import flask
from flask import Flask, render_template, request, jsonify
import pyautogui
import webbrowser
import time
import pyperclip
import threading

app = Flask(__name__)

# Function that will run the automation script
def send_whatsapp_messages(number, message, repeat_count):
    # Open WhatsApp Web with target chat
    url = f"https://web.whatsapp.com/send?phone={number}&text="
    webbrowser.open(url)

    # Wait for WhatsApp Web to load (adjust if needed)
    print("Loading WhatsApp Web... Please wait 15 seconds.")
    time.sleep(15)

    # Send messages very fast
    for i in range(repeat_count):
        pyperclip.copy(message)  # Copy the original message
        pyautogui.hotkey("ctrl", "v")  # Paste the message
        pyautogui.press("enter")  # Press Enter to send
        print(f"Sent: {message}")  # Print out the message sent
        time.sleep(0.1)  # Super fast! Can reduce to 0.1 if stable

    return {"status": "success", "messages_sent": repeat_count}

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/send', methods=['POST'])
def send():
    data = request.json
    number = data.get('phone', '').strip()
    message = data.get('message', '').strip()
    repeat_count = int(data.get('count', 1))

    # Start the sending process in a separate thread
    # This allows the server to respond immediately while the automation runs
```

```
    thread = threading.Thread(
        target=send_whatsapp_messages,
        args=(number, message, repeat_count)
    )
    thread.daemon = True
    thread.start()

    return jsonify({"status": "started", "message": "WhatsApp Web is opening. Please wait."})

if __name__ == '__main__':
    print("Starting WhatsApp Automation Server...")
    print("Open your browser and navigate to http://127.0.0.1:5000")
    app.run(debug=True)
```

**In templates/index.html:**

```
html
```

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>WhatsApp Message Sender</title>
    <style>
        body {
            font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
            background-color: #f0f2f5;
            margin: 0;
            padding: 20px;
            display: flex;
            justify-content: center;
            align-items: center;
            min-height: 100vh;
        }
        .container {
            background-color: white;
            border-radius: 10px;
            box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
            width: 100%;
            max-width: 500px;
            padding: 30px;
        }
        h1 {
            color: #128C7E;
            margin-top: 0;
            text-align: center;
        }
        .form-group {
            margin-bottom: 20px;
        }
        label {
            display: block;
            margin-bottom: 8px;
            font-weight: 500;
            color: #333;
        }
        input, textarea {
            width: 100%;
            padding: 12px;
            border: 1px solid #ddd;
```

```
            border-radius: 6px;
            box-sizing: border-box;
            font-size: 16px;
        }
        textarea {
            min-height: 100px;
            resize: vertical;
        }
        button {
            background-color: #128C7E;
            color: white;
            border: none;
            padding: 12px 20px;
            border-radius: 6px;
            cursor: pointer;
            font-size: 16px;
            width: 100%;
            font-weight: 500;
            transition: background-color 0.3s;
        }
        button:hover {
            background-color: #075E54;
        }
        .status {
            margin-top: 20px;
            padding: 15px;
            border-radius: 6px;
            background-color: #f8f9fa;
            display: none;
        }
        .status.success {
            display: block;
            background-color: #d4edda;
            color: #155724;
            border: 1px solid #c3e6cb;
        }
        .status.error {
            display: block;
            background-color: #f8d7da;
            color: #721c24;
            border: 1px solid #f5c6cb;
        }
        .status.warning {
            display: block;
```

```
            background-color: #fff3cd;
            color: #856404;
            border: 1px solid #ffeeba;
        }
        .status.info {
            display: block;
            background-color: #d1ecf1;
            color: #0c5460;
            border: 1px solid #bee5eb;
        }
        .instructions {
            margin-top: 25px;
            background-color: #e9f7fe;
            padding: 15px;
            border-radius: 6px;
            font-size: 14px;
        }
        .instructions h3 {
            margin-top: 0;
            color: #0277bd;
        }
        .instructions ol {
            padding-left: 20px;
            margin-bottom: 0;
        }
        .instructions li {
            margin-bottom: 8px;
        }
    </style>
</head>
<body>
    <div class="container">
        <h1>WhatsApp Message Sender</h1>

        <div class="form-group">
            <label for="phone">Phone Number (with country code)</label>
            <input type="text" id="phone" placeholder="e.g. 91XXXXXXXXXX" required>
        </div>

        <div class="form-group">
            <label for="message">Message</label>
            <textarea id="message" placeholder="Type your message here..." required></textarea>
        </div>
```

```
        <div class="form-group">
            <label for="count">Number of Times to Send</label>
            <input type="number" id="count" min="1" max="1000" value="1" required>
        </div>

        <button onclick="sendMessage()">Start Sending</button>

        <div id="status" class="status"></div>

        <div class="instructions">
            <h3>Important Instructions:</h3>
            <ol>
                <li>Enter the recipient's phone number with country code (no + symbol)</li>
                <li>Type your message</li>
                <li>Choose how many times to send the message</li>
                <li>Click "Start Sending"</li>
                <li>WhatsApp Web will open automatically</li>
                <li><strong>Do not touch your mouse or keyboard</strong> during the automation
                <li>Make sure you're already logged into WhatsApp Web or be ready to scan the Q
            </ol>
        </div>
    </div>

    <script>
        function sendMessage() {
            const phone = document.getElementById('phone').value.trim();
            const message = document.getElementById('message').value.trim();
            const count = document.getElementById('count').value;
            const statusDiv = document.getElementById('status');

            if (!phone || !message || !count) {
                statusDiv.className = "status error";
                statusDiv.textContent = "Please fill in all fields";
                return;
            }

            if (isNaN(count) || count < 1) {
                statusDiv.className = "status error";
                statusDiv.textContent = "Please enter a valid number of messages";
                return;
            }

            // Prepare status message
            statusDiv.className = "status info";
```

```
            statusDiv.textContent = "Starting WhatsApp Web... Please wait and don't touch your

            // Send the request to the server
            fetch('/send', {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/json',
                },
                body: JSON.stringify({
                    phone: phone,
                    message: message,
                    count: count
                }),
            })
            .then(response => response.json())
            .then(data => {
                statusDiv.className = "status warning";
                statusDiv.textContent = "WhatsApp Web is opening. Do not touch your mouse or ke
            })
            .catch((error) => {
                statusDiv.className = "status error";
                statusDiv.textContent = "Error: " + error.message;
            });
        }
    </script>
</body>
</html>
```

**In requirements.txt:**

```
flask==2.0.1
pyautogui==0.9.53
pyperclip==1.8.2
```

**In README.md:**

```markdown
# WhatsApp Automation Web App

A web application for automating WhatsApp message sending. This application provides a user-fri

## Features

- Send the same message multiple times to a WhatsApp contact
- User-friendly web interface
- Super fast message sending

## Installation

1. Clone this repository
```

git clone https://github.com/YOUR_USERNAME/whatsapp-automation-app.git
cd whatsapp-automation-app

```
2. Install required packages
```

pip install -r requirements.txt

```
3. Run the application
```

python app.py

```
4. Open your web browser and navigate to http://127.0.0.1:5000

## Usage Instructions

1. Enter the recipient's phone number with country code (no + symbol)
2. Type your message
3. Choose how many times to send the message
4. Click "Start Sending"
5. WhatsApp Web will open automatically
6. Do not touch your mouse or keyboard during the automation process
7. Make sure you're already logged into WhatsApp Web or be ready to scan the QR code when prompted

## Important Notes

- This application runs on your local machine only - it's not a hosted service
- The application uses PyAutoGUI to automate mouse and keyboard actions
- Don't interact with your computer while the automation is running
- Use responsibly and respect WhatsApp's terms of service

## Disclaimer

This tool is for educational purposes only. Mass messaging on WhatsApp may violate their terms of service. Use responsibly and at your own risk.
```

## Step 5: Commit and Push to GitHub

1. Add all files to Git:

```
git add .
```

2. Commit the changes:

```
git commit -m "Initial commit of WhatsApp Automation App"
```

3. Pull the README.md from GitHub (to avoid conflicts):

```
git pull origin main --allow-unrelated-histories
```

4. Push your code to GitHub:

```
git push -u origin main
```

## Step 6: Hosting Options

Since GitHub doesn't directly host Python web applications, here are some options for actually running this app online:

### Option 1: Railway.app

1. Create an account at Railway.app
2. Connect your GitHub repository
3. Set up a new project from your GitHub repo
4. Note: PyAutoGUI might face issues in cloud environments as it requires a desktop environment

### Option 2: Heroku

1. Create an account at Heroku
2. Install the Heroku CLI
3. Add a Procfile to your project with the content: `web: gunicorn app:app`
4. Add gunicorn to requirements.txt
5. Note: Same limitation with PyAutoGUI applies

### Option 3: PythonAnywhere

1. Create an account at PythonAnywhere
2. Upload your files or clone from GitHub
3. Set up a web app with Flask
4. Note: Same limitation with PyAutoGUI applies

## Important Note About Cloud Hosting

**Most cloud hosting platforms do not support PyAutoGUI and screen automation because they don't have graphical environments**. Your app is designed to control a local browser and interact with WhatsApp Web, which requires:

1. A graphical environment (desktop)
2. A web browser
3. The ability to control mouse and keyboard

For these reasons, **this application is best run locally on your computer**, not hosted on a cloud service. The GitHub repository serves primarily as a way to share and distribute the code.