🚀 FLOWISE proxy and UI Integration Guide

📋 Current Setup

- **Flowise**: Running at `https://project-1-13.eduhk.hk/`
- **New Proxy Frontend**: Will run on `localhost:5002` → Accessible at `https://project-1-13.eduhk.hk/projectproxy/`
- **New Proxy API**: Will run on `localhost:5000` → API at `https://project-1-13.eduhk.hk/projectproxy/chat/`

🔧 Step 1: Update Nginx Configuration

Add to Existing Config

Edit your current nginx config file:

```
git clone https://github.com/enoch-sit/flowiseProjectProxy.git
sudo nano /etc/nginx/sites-available/$HOSTNAME
```

Add these location blocks **BEFORE** the root `location /` block:

```
# Handle /projectproxy without trailing slash - redirect to /projectproxy/
location = /projectproxy {
    return 301 /projectproxy/;
}

# Minimal Flowise Proxy Frontend (Port 5002)
location /projectproxy/ {
    proxy_pass http://localhost:5002/;
    proxy_set_header Host $host;
    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;
}

# Minimal Flowise Proxy API (Port 5000)
location /projectproxy/chat/ {
    proxy_pass http://localhost:5000/projectproxy/chat/;
    proxy_set_header Host $host;
    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;

    # Enable streaming for Server-Sent Events
    proxy_buffering off;
    proxy_cache off;
    proxy_read_timeout 300s;
```

```
    # Handle CORS
    add_header Access-Control-Allow-Origin '*' always;
    add_header Access-Control-Allow-Methods 'GET, POST, OPTIONS' always;
    add_header Access-Control-Allow-Headers 'Content-Type, Authorization' always;

    if ($request_method = 'OPTIONS') {
        add_header Access-Control-Allow-Origin '*';
        add_header Access-Control-Allow-Methods 'GET, POST, OPTIONS';
        add_header Access-Control-Allow-Headers 'Content-Type, Authorization';
        add_header Content-Length 0;
        add_header Content-Type 'text/plain';
        return 204;
    }
}
```

🔄 Step 2: Test and Reload Nginx

1. **Test Configuration:**

```
sudo nginx -t
```

2. **Reload Nginx:**

```
sudo systemctl reload nginx
```

🚀 Step 3: Initial Service Setup

1. **Set Up Virtual Environment (Recommended):**

Install Python Virtual Environment Package (if needed):

```
# For Ubuntu/Debian systems - install venv package first
sudo apt update
sudo apt install python3-venv -y
```

Create and Activate Virtual Environment:

```
cd ~/flowiseProjectProxy

# Create virtual environment
python3 -m venv venv

# If you get "ensurepip is not available" error, run:
```

```
# sudo apt install python3.12-venv (or your Python version)
# Then recreate: rm -rf venv && python3 -m venv venv

# Activate virtual environment
source venv/bin/activate

# Verify activation (should show venv path)
which python
```

2. **Install Dependencies:**

```
# With virtual environment activated
cd backend
pip install -r requirements.txt
cd ..
```

3. **Configure Environment Variables:**

```
# Create the .env configuration file
cd backend
nano .env
```

Add this content to the .env file:

```
# Flowise Configuration
# IMPORTANT: Choose the correct URL based on your setup
# Option 1: If proxy runs on SAME server as Flowise (recommended)
FLOWISE_API_URL=http://localhost:3000

# Option 2: If proxy runs on DIFFERENT server than Flowise
# FLOWISE_API_URL=https://project-1-13.eduhk.hk

FLOWISE_API_KEY=

# Server Configuration
HOST=0.0.0.0
PORT=5000
BASE_PATH=/projectproxy
```

Save and exit nano:

- Press `Ctrl+X`
- Press `Y` to confirm save
- Press `Enter` to confirm filename

Verify the file:

```
cat .env
cd ..
```

Important Notes:

- **FLOWISE_API_KEY**: Leave empty if your Flowise instance doesn't require authentication.
- **If Flowise requires API key**: Get it from your Flowise dashboard and replace the empty value.
- **BASE_PATH**: Must match the nginx location path (`/projectproxy`).

⚙️ Service Management

Once the initial setup is complete, you can manage the backend and frontend services using one of the methods below.

Method 1: Using the Management Script (Recommended)

The `manage.sh` script in the `maintain/` directory simplifies managing your services.

1. Make the script executable (run once):

```
sudo chmod +x maintain/manage.sh
```

2. Use the script:

```
# Navigate to your project directory
cd ~/flowiseProjectProxy

# Start both services in the background
./maintain/manage.sh start

# Stop both services
./maintain/manage.sh stop

# Restart both services
./maintain/manage.sh restart

# Check the running status of the services
./maintain/manage.sh status

# View the live logs for both services
./maintain/manage.sh logs
```

Method 2: Manual Management

If you prefer not to use the script, you can run the commands manually.

1. Start Services in Background:

```
# Navigate to your project directory
cd ~/flowiseProjectProxy

# Activate virtual environment
source venv/bin/activate

# Create logs directory
mkdir -p logs

# Start backend in background
cd backend
nohup python -m uvicorn main:app --host 0.0.0.0 --port 5000 > ../logs/backend.log
2>&1 &
echo $! > ../backend.pid
cd ..

# Start frontend in background
nohup python frontend_server.py > logs/frontend.log 2>&1 &
echo $! > frontend.pid

echo "☑ Services running in background. Use 'ps aux' to verify."
```

2. Stop Services:

Option A: Using PID files (created by the start commands above)

```
# Navigate to your project directory
cd ~/flowiseProjectProxy

# Stop services
kill $(cat backend.pid)
kill $(cat frontend.pid)
rm backend.pid frontend.pid
```

Option B: Force stop if PID files are missing

```
pkill -f "uvicorn main:app"
pkill -f "frontend_server.py"
```

3. Check Service Status:

```
# Check if ports are listening
sudo apt install -y net-tools
netstat -tuln | grep -E '5000|5002'
```

```
# Or check running processes
ps aux | grep -E 'uvicorn|frontend_server'
```

4. View Logs:

```
# Navigate to your project directory
cd ~/flowiseProjectProxy

# View backend logs
tail -f logs/backend.log

# View frontend logs
tail -f logs/frontend.log
```

When to Restart Services

- **Changed `.env` file**: Restart the **backend**.
- **Changed `backend/main.py`**: Restart the **backend**.
- **Changed `frontend_server.py`**: Restart the **frontend**.
- **Changed `frontend/index.html`**: No restart needed (it's a static file).
- **Installed new Python packages**: Restart **both** services.

🌐 Access Points

After setup, you'll have:

Main Flowise (Existing)

- **URL**: `https://project-1-13.eduhk.hk/`
- **Purpose**: Your original Flowise instance

Minimal Proxy Chat Interface (New)

- **Production URL**: `https://project-1-13.eduhk.hk/projectproxy/`
- **Local URL**: `https://project-1-13/projectproxy/` (if using local domain)
- **Purpose**: Simple chat interface for testing
- **Features**: Clean UI, streaming responses, chatflow ID configuration

Proxy API (New)

- **Production URL**: `https://project-1-13.eduhk.hk/projectproxy/chat/stream`
- **Local URL**: `https://project-1-13/projectproxy/chat/stream`
- **Purpose**: API endpoint for chat requests
- **Docs**: `https://project-1-13.eduhk.hk/projectproxy/docs`

🔧 Configuration

The services are configured with:

- **Basepath**: `/projectproxy` (matches nginx routing)
- **Flowise Connection**: `https://project-1-13.eduhk.hk`
- **CORS**: Enabled for cross-origin requests
- **Streaming**: Server-Sent Events for real-time responses

🔧 Troubleshooting

1. **502 Bad Gateway**: Check if services are running on ports 5000 and 5002
2. **CORS Errors**: Verify nginx CORS headers are properly configured
3. **No Response**: Ensure Flowise is accessible at `https://project-1-13.eduhk.hk`
4. **Permission Denied**: Check file permissions and user access
5. **Nginx Config Error**: `location "/projectproxy" is outside location "/projectproxy/"`
    - **Fix**: Use the updated config with separate `location = /projectproxy` block
    - **Root Cause**: Nested location blocks are not allowed in nginx
    - **Solution**: Use `nginx-fixed-config.conf` instead of the old version
6. **Python Package Conflicts**: Different projects require different package versions
    - **Fix**: Use virtual environment (see `VIRTUAL_ENV_SETUP.md`)
    - **Command**: `python3 -m venv venv && source venv/bin/activate`
    - **Benefits**: Isolated dependencies, easy cleanup, professional practice
7. **Virtual Environment Creation Fails**: `ensurepip is not available`
    - **Error**: `The virtual environment was not created successfully because ensurepip is not available`
    - **Ubuntu/Debian Fix**: `sudo apt install python3-venv` or `sudo apt install python3.12-venv`
    - **CentOS/RHEL Fix**: `sudo yum install python3-venv` or `sudo dnf install python3-venv`
    - **After Installation**: `rm -rf venv && python3 -m venv venv`
8. **404 Not Found on /projectproxy/chat/stream**: Backend endpoint not found
    - **Root Cause**: Missing or incorrect `.env` configuration file
    - **Error in logs**: `"POST /projectproxy/chat/stream HTTP/1.0" 404 Not Found`
    - **Fix**: Create `backend/.env` file with correct `BASE_PATH=/projectproxy`
    - **Commands**: See "Step 1.5: Configure Environment Variables" above
    - **After Fix**: Restart backend service
9. **Wrong Flowise Connection URL**: Choose localhost vs external URL
    - **Test Method**: Run `python test_flowise_connection.py` in your project folder
    - **Same Server Setup** (recommended): Use `FLOWISE_API_URL=http://localhost:3000`
    - **Different Server Setup**: Use `FLOWISE_API_URL=https://project-1-13.eduhk.hk`
    - **Performance**: Localhost is faster for same-server deployment
    - **Signs of Wrong URL**: Connection timeouts, 502 errors, or slow responses

📝 Testing

Production Testing

1. Visit `https://project-1-13.eduhk.hk/projectproxy/`
2. Enter your Chatflow ID from Flowise dashboard
3. Send a test message

4. Verify streaming response works

Local Testing (if using local domain)

1. Visit `https://project-1-13/projectproxy/`
2. Follow the same steps as above

This setup allows you to have both your original Flowise instance and the minimal proxy running simultaneously!