**Frontend (Node.js with Express)**

**Project Structure**

pgsql

CopyEdit

my-express-app/

├── public/

│ ├── index.html

│ ├── style.css

│ └── script.js

├── server.js

├── package.json

**1. Initialize the Project**

bash

CopyEdit

mkdir my-express-app

cd my-express-app

npm init -y

npm install express

**2. server.js – Basic Express Server**

js

CopyEdit

const express = require('express');

const path = require('path');

const app = express();

const PORT = 3000;

// Serve static files from the "public" directory

app.use(express.static(path.join(\_\_dirname, 'public')));

// Default route

app.get('/', (req, res) => {

res.sendFile(path.join(\_\_dirname, 'public', 'index.html'));

});

app.listen(PORT, () => {

console.log(`Server running at http://localhost:${PORT}`);

});

**3. public/index.html**

html

CopyEdit

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>Express Frontend</title>

<link rel="stylesheet" href="style.css" />

</head>

<body>

<h1>Hello from Express Frontend!</h1>

<button onclick="showMessage()">Click Me</button>

<p id="message"></p>

<script src="script.js"></script>

</body>

</html>

**4. public/style.css**

css

CopyEdit

body {

font-family: Arial, sans-serif;

padding: 2rem;

background-color: #f0f0f0;

}

h1 {

color: #333;

}

**5. public/script.js**

js

CopyEdit

function showMessage() {

document.getElementById('message').textContent = 'You clicked the button!';

}

**6. Run the App**

bash

CopyEdit

node server.js

Open your browser and go to <http://localhost:3000>

**Updated Project Structure**

pgsql

CopyEdit

my-express-app/

├── public/

│ ├── index.html

│ ├── style.css

│ └── script.js

├── server.js

├── package.json

**📄 1. public/index.html – Add a Form**

html

CopyEdit

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>Express Form Example</title>

<link rel="stylesheet" href="style.css" />

</head>

<body>

<h1>Contact Form</h1>

<form id="contactForm" action="/submit" method="POST">

<label for="name">Name:</label><br />

<input type="text" id="name" name="name" required /><br /><br />

<label for="email">Email:</label><br />

<input type="email" id="email" name="email" required /><br /><br />

<label for="message">Message:</label><br />

<textarea id="message" name="message" required></textarea><br /><br />

<button type="submit">Submit</button>

</form>

</body>

</html>

**📄 2. server.js – Handle Form POST Request**

js

CopyEdit

const express = require('express');

const path = require('path');

const app = express();

const PORT = 3000;

// Middleware to parse form data

app.use(express.urlencoded({ extended: true }));

// Serve static files

app.use(express.static(path.join(\_\_dirname, 'public')));

// Serve HTML form

app.get('/', (req, res) => {

res.sendFile(path.join(\_\_dirname, 'public', 'index.html'));

});

// Handle form submission

app.post('/submit', (req, res) => {

const { name, email, message } = req.body;

console.log('Form submission received:');

console.log(`Name: ${name}`);

console.log(`Email: ${email}`);

console.log(`Message: ${message}`);

// Response for the user

res.send(`

<h1>Thank You!</h1>

<p><strong>Name:</strong> ${name}</p>

<p><strong>Email:</strong> ${email}</p>

<p><strong>Message:</strong> ${message}</p>

<a href="/">Go Back</a>

`);

});

// Start server

app.listen(PORT, () => {

console.log(`Server is running at http://localhost:${PORT}`);

});

**🟦 3. style.css – Optional Styling**

css

CopyEdit

body {

font-family: Arial, sans-serif;

padding: 2rem;

background-color: #f9f9f9;

}

form {

background-color: #fff;

padding: 1rem;

border: 1px solid #ddd;

max-width: 400px;

}

input, textarea {

width: 100%;

padding: 0.5rem;

margin-top: 0.3rem;

}

**▶ Run It**

bash

CopyEdit

node server.js

Visit <http://localhost:3000>, fill in the form, and submit. You’ll see the form data echoed back.

1. Make the HTML form **send a POST request to the Flask server**.
2. Allow **CORS** on your Flask backend so it accepts requests from the Express frontend.
3. Use **JavaScript (fetch)** instead of traditional form submission so it’s seamless and cross-origin-friendly.

**✅ Here's how to set it up:**

**💻 Step 1: Frontend (Express) – Update index.html**

Update your form to use JavaScript to POST data to the Flask backend.

**public/index.html**

html

CopyEdit

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0"/>

<title>Express Form to Flask</title>

<link rel="stylesheet" href="style.css" />

</head>

<body>

<h1>Contact Form</h1>

<form id="contactForm">

<label for="name">Name:</label><br />

<input type="text" id="name" name="name" required /><br /><br />

<label for="email">Email:</label><br />

<input type="email" id="email" name="email" required /><br /><br />

<label for="message">Message:</label><br />

<textarea id="message" name="message" required></textarea><br /><br />

<button type="submit">Submit</button>

</form>

<p id="responseMsg"></p>

<script>

const form = document.getElementById('contactForm');

const responseMsg = document.getElementById('responseMsg');

form.addEventListener('submit', async (e) => {

e.preventDefault();

const data = {

name: form.name.value,

email: form.email.value,

message: form.message.value

};

try {

const res = await fetch('http://127.0.0.1:5000/submit', {

method: 'POST',

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify(data)

});

const result = await res.json();

responseMsg.textContent = result.message;

} catch (err) {

responseMsg.textContent = 'Error submitting the form.';

console.error(err);

}

});

</script>

</body>

</html>

**🐍 Step 2: Flask Backend – Create app.py**

Here’s a simple Flask app that receives JSON data from the Express frontend.

**app.py**

python

CopyEdit

from flask import Flask, request, jsonify

from flask\_cors import CORS

app = Flask(\_\_name\_\_)

CORS(app) # Allow requests from other origins

@app.route('/submit', methods=['POST'])

def submit():

data = request.get\_json()

name = data.get('name')

email = data.get('email')

message = data.get('message')

print(f"Received from frontend:\nName: {name}\nEmail: {email}\nMessage: {message}")

return jsonify({

'message': f'Thank you, {name}! Your message has been received.'

})

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

**🛠️ Step 3: Run Both Servers**

* Start **Flask backend**:

bash

CopyEdit

python app.py

It will run on http://127.0.0.1:5000

* Start **Express frontend**:

bash

CopyEdit

node server.js

It will serve your static frontend at http://localhost:3000

**🔄 Test It**

1. Open http://localhost:3000
2. Fill out the form
3. Click "Submit"
4. Flask backend logs the submission
5. The page shows a confirmation message

**Backend (Flask)**

**1. Adjust the Form in Express to Submit to Flask Directly**

Update your Express form.ejs so that the <form> action points directly to the Flask backend.

**views/form.ejs**

html

CopyEdit

<!DOCTYPE html>

<html>

<head>

<title>Express Form</title>

</head>

<body>

<h1>Submit Your Info</h1>

<form action="http://localhost:5000/submit-form" method="POST">

<label for="name">Name:</label>

<input type="text" id="name" name="name" required><br>

<label for="email">Email:</label>

<input type="email" id="email" name="email" required><br>

<label for="message">Message:</label>

<textarea id="message" name="message" required></textarea><br>

<button type="submit">Submit</button>

</form>

</body>

</html>

* This form **bypasses Express’s POST route** and sends data **directly to your Flask backend** at http://localhost:5000/submit-form.

**✅ 2. Flask Backend to Handle Form Submission**

Here’s a simple Flask app that accepts the form data, processes it, and returns a response.

**app.py (Flask backend)**

python

CopyEdit

from flask import Flask, request, render\_template\_string

app = Flask(\_\_name\_\_)

@app.route('/submit-form', methods=['POST'])

def handle\_form():

name = request.form.get('name')

email = request.form.get('email')

message = request.form.get('message')

# Example processing: Print to console or save to a file/db

print(f"Received: Name={name}, Email={email}, Message={message}")

# Respond with a basic confirmation page

return render\_template\_string("""

<h1>Thank you, {{ name }}!</h1>

<p>We received your message:</p>

<blockquote>{{ message }}</blockquote>

<p>We'll respond to {{ email }} soon.</p>

<a href="http://localhost:3000">Back to form</a>

""", name=name, email=email, message=message)

if \_\_name\_\_ == '\_\_main\_\_':

app.run(port=5000)

**✅ Summary**

| **Component** | **Role** |
| --- | --- |
| **Express frontend (Node.js)** | Renders the HTML form |
| **Flask backend (Python)** | Handles the POST data, processes it, and sends a response |

**🛡️ Optional: Cross-Origin Concerns**

If your form sends data via JavaScript fetch() or you're dealing with a different domain (not just localhost), you may need to configure **CORS** in Flask:

bash

CopyEdit

pip install flask-cors

python

CopyEdit

from flask\_cors import CORS

CORS(app)

**Folder Structure**

**Project Structure**

csharp

CopyEdit

project-root/

├── backend/ # Flask backend

│ ├── app.py

│ ├── requirements.txt

│ └── templates/ # Optional: Flask templates

│ └── response.html

│

├── frontend/ # Express frontend

│ ├── app.js

│ ├── package.json

│ ├── views/

│ │ └── form.ejs

│ └── public/

│ └── style.css

**✅ Step-by-Step Setup**

**🔧 1. Backend (backend/app.py) – Flask**

python

CopyEdit

from flask import Flask, request, render\_template

app = Flask(\_\_name\_\_)

@app.route('/submit-form', methods=['POST'])

def handle\_form():

name = request.form.get('name')

email = request.form.get('email')

message = request.form.get('message')

return render\_template('response.html', name=name, email=email, message=message)

if \_\_name\_\_ == '\_\_main\_\_':

app.run(port=5000)

**📝 backend/templates/response.html**

html

CopyEdit

<!DOCTYPE html>

<html>

<head>

<title>Form Submitted</title>

</head>

<body>

<h1>Thank you, {{ name }}!</h1>

<p>Email: {{ email }}</p>

<p>Your message:</p>

<blockquote>{{ message }}</blockquote>

<a href="http://localhost:3000">Back to Form</a>

</body>

</html>

**📦 backend/requirements.txt**

nginx

CopyEdit

Flask

Install dependencies:

bash

CopyEdit

cd backend

pip install -r requirements.txt

**🔧 2. Frontend (frontend/app.js) – Node.js/Express**

js

CopyEdit

const express = require('express');

const path = require('path');

const app = express();

app.set('view engine', 'ejs');

app.set('views', path.join(\_\_dirname, 'views'));

app.use(express.static(path.join(\_\_dirname, 'public')));

// Render form

app.get('/', (req, res) => {

res.render('form');

});

const PORT = 3000;

app.listen(PORT, () => {

console.log(`Frontend running at http://localhost:${PORT}`);

});

**📄 frontend/views/form.ejs**

html

CopyEdit

<!DOCTYPE html>

<html>

<head>

<title>Submit Info</title>

</head>

<body>

<h1>Submit Your Info</h1>

<form action="http://localhost:5000/submit-form" method="POST">

<label>Name:</label>

<input type="text" name="name" required><br>

<label>Email:</label>

<input type="email" name="email" required><br>

<label>Message:</label>

<textarea name="message" required></textarea><br>

<button type="submit">Submit</button>

</form>

</body>

</html>

**📦 frontend/package.json**

Generate it using:

bash

CopyEdit

cd frontend

npm init -y

npm install express ejs

**✅ How to Run It**

In **separate terminals**, run the two servers:

**🔁 Flask Backend (Port 5000)**

bash

CopyEdit

cd backend

python app.py

**🔁 Node.js Frontend (Port 3000)**

bash

CopyEdit

cd frontend

node app.js

**✅ Done!**

* Visit the form: <http://localhost:3000>
* Upon submission, the form data is sent to the **Flask backend** (http://localhost:5000/submit-form).
* Flask processes and responds with a confirmation page.

**Docker Configuration**

**Final Project Structure (with Docker support)**

java

CopyEdit

project-root/

├── backend/

│ ├── app.py

│ ├── requirements.txt

│ ├── templates/

│ │ └── response.html

│ └── Dockerfile

│

├── frontend/

│ ├── app.js

│ ├── package.json

│ ├── views/

│ │ └── form.ejs

│ ├── public/

│ │ └── style.css

│ └── Dockerfile

│

└── docker-compose.yml

**🐳 1. backend/Dockerfile (Flask)**

Dockerfile

CopyEdit

# Use official Python image

FROM python:3.11-slim

# Set working directory

WORKDIR /app

# Copy files

COPY . .

# Install dependencies

RUN pip install --no-cache-dir -r requirements.txt

# Expose Flask port

EXPOSE 5000

# Run the Flask app

CMD ["python", "app.py"]

**🐳 2. frontend/Dockerfile (Node.js/Express)**

Dockerfile

CopyEdit

# Use Node.js image

FROM node:20-alpine

# Set working directory

WORKDIR /app

# Copy dependency files and install

COPY package\*.json ./

RUN npm install

# Copy source code

COPY . .

# Expose Express port

EXPOSE 3000

# Start the app

CMD ["node", "app.js"]

**🛠️ 3. docker-compose.yml (To Run Both Together)**

Create this at the root of your project:

yaml

CopyEdit

version: '3.9'

services:

backend:

build: ./backend

ports:

- "5000:5000"

restart: always

frontend:

build: ./frontend

ports:

- "3000:3000"

restart: always

**🚀 Build and Run the Containers**

In the root of your project:

bash

CopyEdit

docker-compose up --build

* Frontend available at: <http://localhost:3000>
* Backend handles POST at: <http://localhost:5000/submit-form>

**🧼 Optional: .dockerignore Files**

**In backend/.dockerignore:**

markdown

CopyEdit

\_\_pycache\_\_/

\*.pyc

\*.pyo

\*.pyd

.env

**In frontend/.dockerignore:**

lua

CopyEdit

node\_modules

npm-debug.log

.env

**docker-compose.yml**

Place this in the **root of your project**:

yaml

CopyEdit

version: '3.9'

services:

backend:

build: ./backend

container\_name: flask-backend

ports:

- "5000:5000"

networks:

- webnet

restart: unless-stopped

frontend:

build: ./frontend

container\_name: express-frontend

ports:

- "3000:3000"

networks:

- webnet

restart: unless-stopped

**✅ Docker Network Definition**

At the bottom of the same file, define the custom network:

yaml

CopyEdit

networks:

webnet:

driver: bridge

**✅ What This Does**

| **Service** | **Hostname (from other containers)** | **Accessible on host** |
| --- | --- | --- |
| backend | flask-backend | <http://localhost:5000> |
| frontend | express-frontend | <http://localhost:3000> |

**✅ Update Frontend to Use Service Name for Backend**

In your **form.ejs**, change the form's action so the frontend container posts to the backend container using its **Docker service name**:

html

CopyEdit

<form action="http://flask-backend:5000/submit-form" method="POST">

✅ This ensures the frontend container can talk to the backend container **inside Docker** using internal networking.

**✅ Build and Run**

In the root of your project:

bash

CopyEdit

docker-compose up --build

To run in the background:

bash

CopyEdit

docker-compose up --build -d

**✅ Accessing the App**

* Open <http://localhost:3000> to view the form.
* Submit it — the Express frontend will post to the Flask backend through Docker network.

**1. Create .gitignore File**

In the **project root**, add a .gitignore file:

**.gitignore**

gitignore

CopyEdit

# Node modules (frontend)

frontend/node\_modules/

frontend/npm-debug.log

# Python cache (backend)

backend/\_\_pycache\_\_/

backend/\*.pyc

backend/\*.pyo

backend/\*.pyd

# VSCode settings

.vscode/

# MacOS & Linux

.DS\_Store

# Environment files

\*.env

**✅ 2. Tag and Push Docker Images to Docker Hub**

Assume your Docker Hub username is toto3107. Replace it with your actual Docker Hub username if it's different.

**🐳 Tag and Push Backend Image**

bash

CopyEdit

# Inside project root

docker build -t toto3107/flaskp-backend ./backend

docker push toto3107/flaskp-backend

**🐳 Tag and Push Frontend Image**

bash

CopyEdit

docker build -t toto3107/flaskp-frontend ./frontend

docker push toto3107/flaskp-frontend

✅ Now your images are available on Docker Hub:

* https://hub.docker.com/r/toto3107/flaskp-backend
* https://hub.docker.com/r/toto3107/flaskp-frontend

**✅ 3. Push to GitHub**

**Steps:**

1. Initialize Git in your root directory:

bash

CopyEdit

git init

1. Add remote repo:

bash

CopyEdit

git remote add origin https://github.com/Toto3107/flaskp.git

1. Stage and commit:

bash

CopyEdit

git add .

git commit -m "Initial commit with Flask backend and Express frontend using Docker"

1. Push to GitHub:

bash

CopyEdit

git branch -M main

git push -u origin main

**✅ Summary**

| **Step** | **Description** |
| --- | --- |
| 🐳 Docker Images | Tagged and pushed: flaskp-frontend, flaskp-backend |
| 📁 GitHub | Code pushed to https://github.com/Toto3107/flaskp |
| ⚙️ Ignored Files | node\_modules/, \_\_pycache\_\_/, .vscode/, etc. |