Create an EmailJS Account

To get the `YOUR\_SERVICE\_ID`, `YOUR\_TEMPLATE\_ID`, and `YOUR\_EMAILJS\_USER\_ID` for EmailJS, follow these steps:

1. \*\*Create an EmailJS Account\*\*:

- Go to [EmailJS](https://www.emailjs.com/) and create an account if you haven't already.

2. \*\*Set Up an Email Service\*\*:

- After logging in, go to the \*\*Email Services\*\* section in your EmailJS dashboard.

- Click on \*\*Add new service\*\* and choose an email provider (e.g., Gmail, Outlook, etc.).

- Follow the setup instructions, and once added, EmailJS will generate a \*\*Service ID\*\* for you. This is your `YOUR\_SERVICE\_ID`.

3. \*\*Create an Email Template\*\*:

- Go to the \*\*Email Templates\*\* section in the dashboard.

- Click on \*\*Create New Template\*\* and customize it with placeholders for dynamic content (e.g., recipient email, message).

- EmailJS will generate a \*\*Template ID\*\* for this template. This is your `YOUR\_TEMPLATE\_ID`.

4. \*\*Get Your User ID\*\*:

- Go to \*\*Account\*\* in the EmailJS dashboard, and you’ll find your \*\*User ID\*\* listed there. This is your `YOUR\_EMAILJS\_USER\_ID`.

5. \*\*Replace the Placeholders\*\*:

- Once you have these values, update your script like so:

```javascript

emailjs.init("YOUR\_EMAILJS\_USER\_ID"); // Replace with your actual User ID

emailjs.send('YOUR\_SERVICE\_ID', 'YOUR\_TEMPLATE\_ID', templateParams)

```

With these values set, your form should be able to send emails via EmailJS!

OR

document.getElementById('sendBtn').addEventListener('click', function(event) {

event.preventDefault(); // Prevent form submission

const recipient = document.getElementById('send\_to\_').value;

const message = document.getElementById('message\_').value;

window.location.href = `mailto:${recipient}?body=${encodeURIComponent(message)}`;

});

Great question! You're on the right track with understanding how EmailJS works.

### Here's how EmailJS works when someone else (other than you) is using the website to send an email:

When you use EmailJS, you're essentially setting up a \*\*pre-configured email template\*\* and specifying a \*\*service\*\* that will send the email on behalf of the user. Even though the \*\*Service ID\*\* and \*\*Template ID\*\* belong to you (since you created the EmailJS account and set them up), the \*\*email is sent by EmailJS on the user's behalf\*\*.

Here’s how it works step by step:

### 1. \*\*Set Up EmailJS (You, the Website Owner)\*\*

- You sign up for an \*\*EmailJS\*\* account.

- You configure an \*\*email service\*\* (like Gmail, SendGrid, etc.) that will send the email.

- You create an \*\*email template\*\* on the EmailJS dashboard. This template defines what the email will look like (subject, body, etc.).

### 2. \*\*Using the Service ID and Template ID\*\*

- \*\*Service ID\*\*: This refers to the specific email service you’re using (e.g., Gmail, SendGrid) and is provided by EmailJS.

- \*\*Template ID\*\*: This is the specific template you created on the EmailJS dashboard, which defines the email content.

- You will include these in your frontend code when you make the request to send an email.

### 3. \*\*User Submits the Form (Someone Else Using Your Website)\*\*

- When a user (someone who is visiting your website) fills out the contact form and clicks "Send", their \*\*form data\*\* (like the recipient's email, message, etc.) gets sent to \*\*EmailJS\*\* using your \*\*Service ID\*\* and \*\*Template ID\*\*.

- The form data is sent to the EmailJS API. The user's data (like the recipient email, the message, etc.) is passed to the \*\*template\*\* you created.

### 4. \*\*Email Sent via EmailJS\*\*

- \*\*EmailJS\*\* uses your \*\*Service ID\*\* and \*\*Template ID\*\* to format and send the email.

- The \*\*email is sent on behalf of the recipient\*\* (which could be you or the email you set in your template).

- You \*\*never need to expose your email credentials or handle the email delivery directly\*\*. EmailJS handles all the delivery in the background.

### How Does EmailJS Know the "To" Address (The Recipient)?

The magic happens in the template itself. You can set up placeholders in your template to accept dynamic values like the recipient’s email address, message body, etc.

For example, in your template, you might have a field like this:

- \*\*To\*\*: `{to\_email}` (This will be replaced with the user’s email input)

- \*\*Message Body\*\*: `{message}` (This will be replaced with the message the user enters)

When you send the request from your website, you replace these placeholders with actual values from the form, like this:

```javascript

const templateParams = {

to\_email: recipient, // Email entered by the user

message: message // Message entered by the user

};

```

### Example Code Flow

1. \*\*User fills the form\*\* on your website with their email and message.

2. \*\*JavaScript sends the data to EmailJS\*\* using your Service ID and Template ID.

3. \*\*EmailJS\*\* uses the template you've created (with placeholders) and sends the email \*\*to the specified recipient\*\* using the configured service.

### Example (Code Implementation)

Here’s how the code might look:

```javascript

document.getElementById('sendBtn').addEventListener('click', function(event) {

event.preventDefault(); // Prevent form submission

const recipient = document.getElementById('send\_to\_').value; // Recipient email

const message = document.getElementById('message\_').value; // Message body

const cSubject = 'My interest in your portfolio'; // Subject

// Prepare template parameters (dynamic data from the form)

const templateParams = {

to\_email: recipient, // The recipient email input by the user

subject: cSubject, // The subject of the email

message: message // The message content entered by the user

};

// Send email via EmailJS API

emailjs.send('YOUR\_SERVICE\_ID', 'YOUR\_TEMPLATE\_ID', templateParams)

.then((response) => {

alert('Email sent successfully!');

}, (error) => {

alert('Failed to send email. Please try again.');

});

});

```

### Key Points:

- \*\*Service ID\*\* and \*\*Template ID\*\*: These are yours because they belong to your EmailJS account.

- \*\*Template Parameters\*\*: These are the dynamic values you send to EmailJS, such as the recipient’s email and the message. These values come from the form input fields that the user fills out.

- \*\*Email Sent Automatically\*\*: EmailJS sends the email \*\*on behalf of the user\*\*, meaning they don't need to interact with an email client like Outlook or Gmail.

### Example Flow for Someone Using Your Website:

1. \*\*Someone\*\* visits your website and enters their email and message in the form.

2. They press \*\*"Send"\*\*, triggering the \*\*EmailJS API call\*\*.

3. \*\*EmailJS\*\* uses the \*\*Service ID\*\* and \*\*Template ID\*\* you configured, replaces the placeholders with the form data, and sends the email on behalf of the recipient.

4. The user sees a success or failure message based on the result.

### Conclusion

- The \*\*Service ID\*\* and \*\*Template ID\*\* belong to you because you're the one setting up EmailJS.

- \*\*Someone else\*\* (the user) can fill out the form and have the email sent \*\*on their behalf\*\*. The email will be sent automatically in the background without them needing to open an email client like Outlook.

- The \*\*user doesn’t need an EmailJS account\*\*—you do the configuration, and they just submit the form.

This setup allows you to collect messages or inquiries from users on your website without requiring them to manually send emails via their own email client.