

Fetch POST PUT DELETE

i need fetch [post , put , delete] on this
const API_URL =
"https://jsonplaceholder.typicode.com/users"; with Async
Await

Sure! Here's how you can use `fetch` with `async/await` for **POST**, **PUT**, and **DELETE** requests using the `API_URL` you provided:

javascript

```
const API_URL = "https://jsonplaceholder.typicode.com/users";

// POST: Create a new user
async function createUser(userData) {
  try {
    const response = await fetch(API_URL, {
      method: "POST",
      headers: {
        "Content-Type": "application/json",
      },
      body: JSON.stringify(userData),
    });

    const data = await response.json();
    console.log("User Created:", data);
  } catch (error) {
    console.error("Error creating user:", error);
  }
}

// PUT: Update an existing user
async function updateUser(userId, updatedData) {
  try {
    const response = await fetch(`${API_URL}/${userId}`, {
      method: "PUT",
      headers: {
        "Content-Type": "application/json",
      },
      body: JSON.stringify(updatedData),
    });
  }
```

```

    const data = await response.json();
    console.log("User Updated:", data);
  } catch (error) {
    console.error("Error updating user:", error);
  }
}

// DELETE: Remove a user
async function deleteUser(userId) {
  try {
    const response = await fetch(`${API_URL}/${userId}`, {
      method: "DELETE",
    });

    if (response.ok) {
      console.log(`User with ID ${userId} deleted successfully`);
    } else {
      console.error("Failed to delete user");
    }
  } catch (error) {
    console.error("Error deleting user:", error);
  }
}

```

Example Usage:

javascript

```

createUser({
  name: "John Doe",
  email: "johndoe@example.com",
});

updateUser(1, {
  name: "Updated Name",
  email: "updated@example.com",
});

deleteUser(1);

```

Let me know if you need the same with `PATCH` or want to wrap this in a class/module.

Perfect! Here's how you can **get data from a form** and then use it in your **POST**, **PUT**, or **DELETE** fetch functions with **async/await**.

Let's assume you have a simple HTML form like this:

html

```
<form id="userForm">
  <input type="text" id="name" name="name" placeholder="Name" required />
  <input type="email" id="email" name="email" placeholder="Email" required />
  <input type="number" id="userId" name="userId" placeholder="User ID (for
update/delete)" />
  <button type="submit">Submit</button>
</form>
```

Now here's the JavaScript:

javascript

```
const API_URL = "https://jsonplaceholder.typicode.com/users";

// Fetch POST
async function createUser(userData) {
  const response = await fetch(API_URL, {
    method: "POST",
    headers: { "Content-Type": "application/json" },
    body: JSON.stringify(userData),
  });
  const data = await response.json();
  console.log("User Created:", data);
}

// Fetch PUT
async function updateUser(userId, updatedData) {
  const response = await fetch(`${API_URL}/${userId}`, {
    method: "PUT",
    headers: { "Content-Type": "application/json" },
    body: JSON.stringify(updatedData),
  });
  const data = await response.json();
  console.log("User Updated:", data);
}
```

```

}

// Fetch DELETE
async function deleteUser(userId) {
  const response = await fetch(`${API_URL}/${userId}`, {
    method: "DELETE",
  });

  if (response.ok) {
    console.log(`User with ID ${userId} deleted`);
  } else {
    console.error("Delete failed");
  }
}

// Form submission handler
document.getElementById("userForm").addEventListener("submit", async function (e) {
  e.preventDefault();

  const name = document.getElementById("name").value;
  const email = document.getElementById("email").value;
  const userId = document.getElementById("userId").value;

  const userData = { name, email };

  if (userId) {
    // If userId exists, update the user
    await updateUser(userId, userData);
  } else {
    // Otherwise, create new user
    await createUser(userData);
  }

  // Optionally reset the form
  e.target.reset();
});

// Optional: Delete user separately
// Example call: deleteUser(1);

```

Notes:

- When `userId` is filled, it triggers **update**.
- When `userId` is empty, it triggers **create**.
- You can add a separate button or flow for deleting by user ID.

Need this turned into a React version? Or want to add validations or feedback messages?

Follow up
