Forms in SSR

Form handling in +page.server.ts

SSR - Form Handling - Donate

- With SSR engaged, we can incorporate form handling in a similar manner to conventional node applications (Hapi)

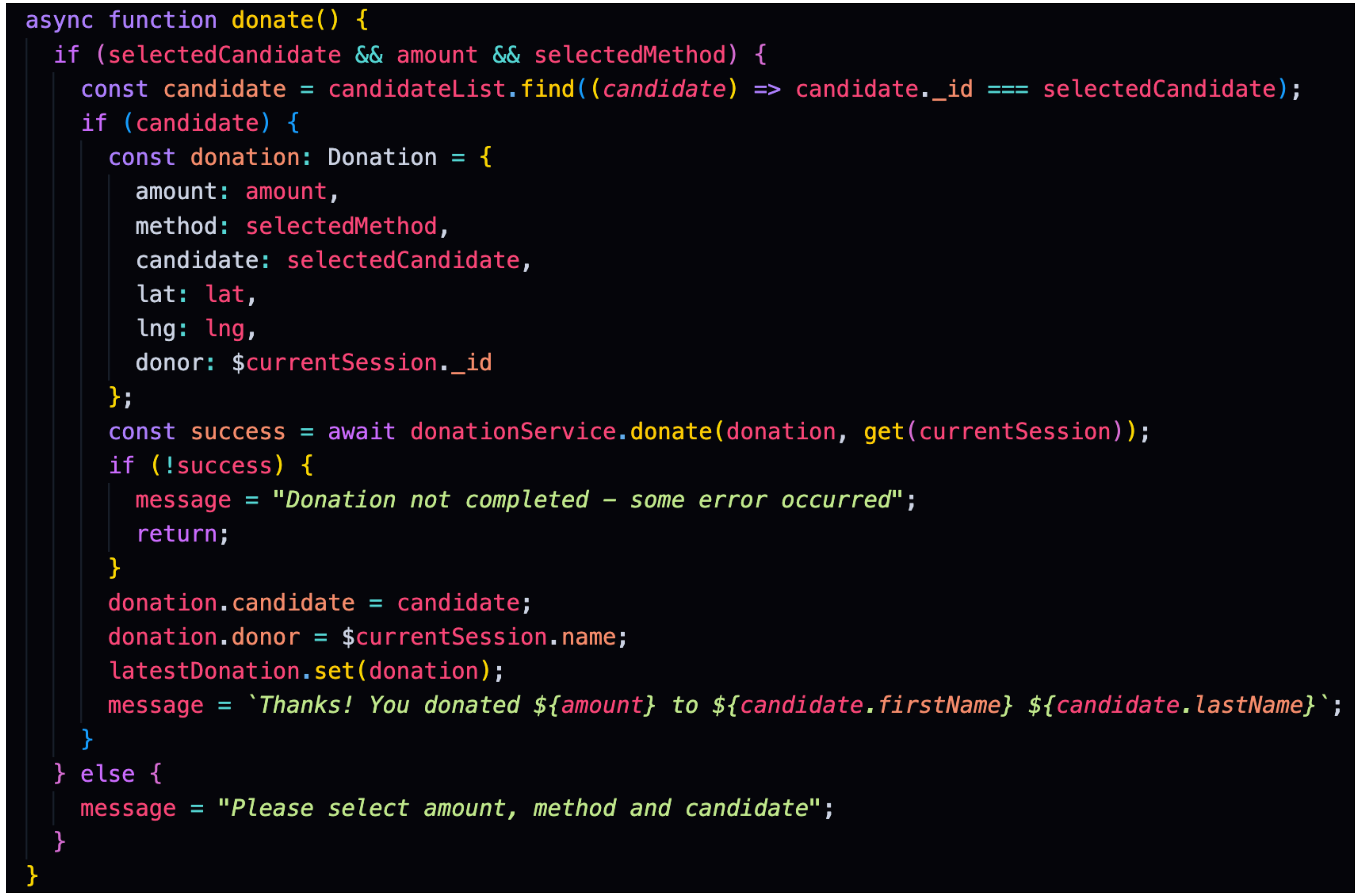
- i.e. the Form data can be ‘Posted’ to the server, and processed there.

- Specifically, this processing can include conventional cookie based session creation + tracking, replacing the store/local storage approach

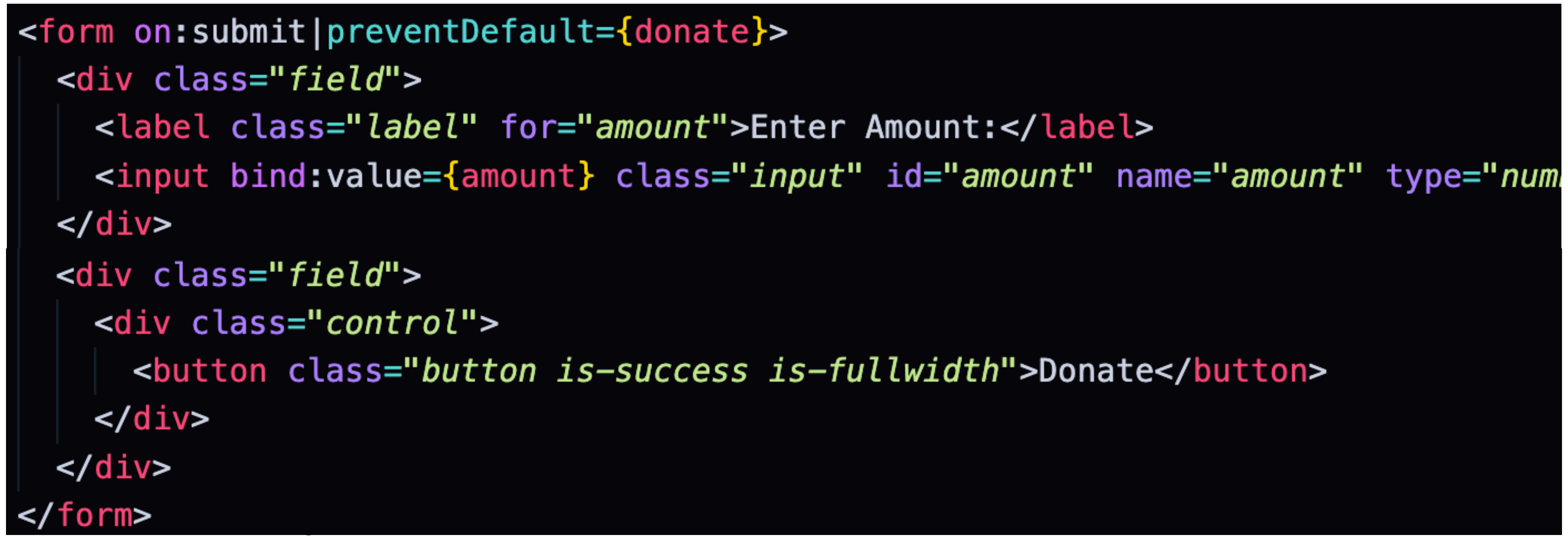
- CSR (Client Side Rendering) approach

- Form button press is intercepted and handled in the browser

- Form data retrieved from the bound variables and sent to the API

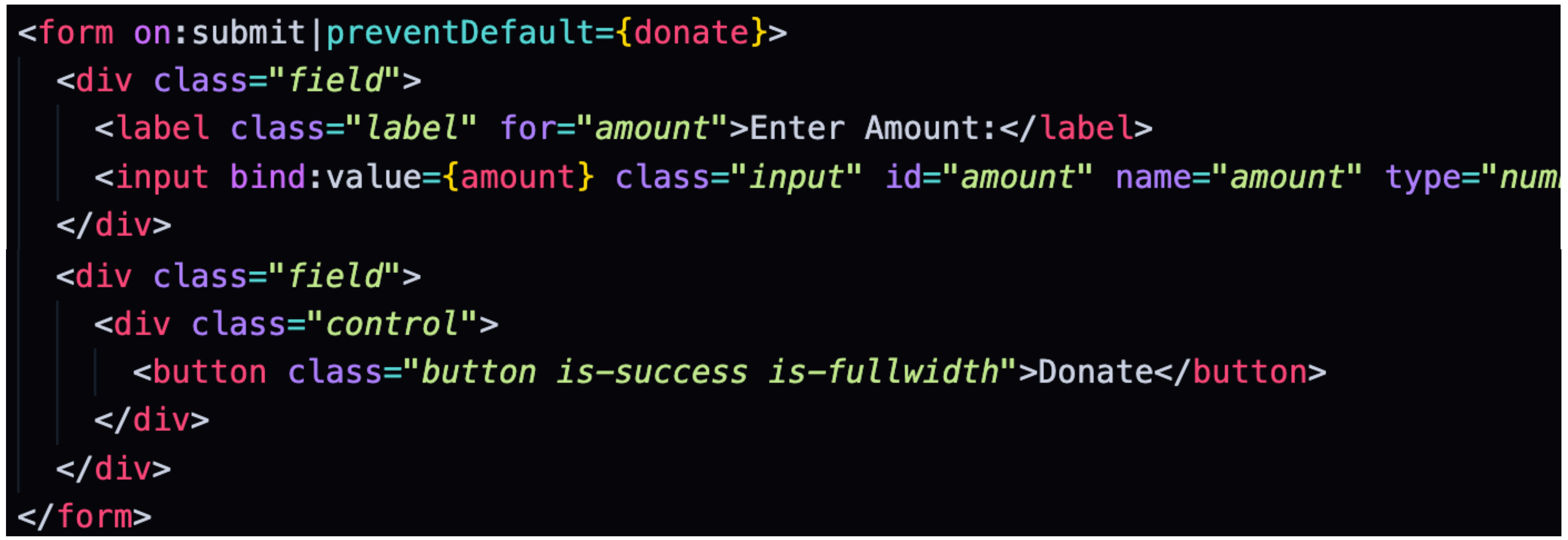


```
async function donate() {
  if (selectedCandidate && amount && selectedMethod) {
    const candidate = candidateList.find((candidate) => candidate._id === selectedCandidate);
    if (candidate) {
      const donation: Donation = {
        amount: amount,
        method: selectedMethod,
        candidate: selectedCandidate,
        lat: lat,
        lng: lng,
        donor: $currentSession._id
      };
      const success = await donationService.donate(donation, get(currentSession));
      if (!success) {
        message = "Donation not completed - some error occurred";
        return;
      }
      donation.candidate = candidate;
      donation.donor = $currentSession.name;
      latestDonation.set(donation);
      message = `Thanks! You donated ${amount} to ${candidate.firstName} ${candidate.lastName}`;
    }
  } else {
    message = "Please select amount, method and candidate";
  }
}
```



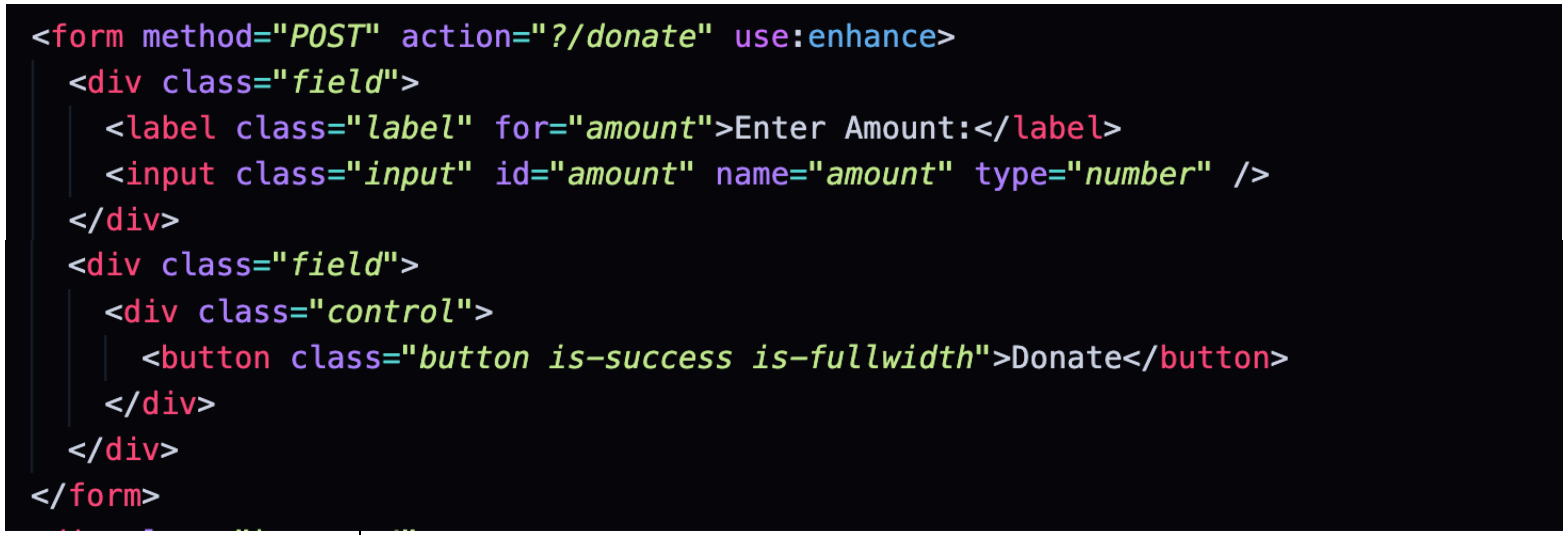
```
<form on:submit|preventDefault={donate}>
  <div class="field">
    <label class="label" for="amount">Enter Amount:</label>
    <input bind:value={amount} class="input" id="amount" name="amount" type="num
  </div>
  <div class="field">
    <div class="control">
      <button class="button is-success is-fullwidth">Donate</button>
    </div>
  </div>
</form>
```

SSR Conversion: Client Side



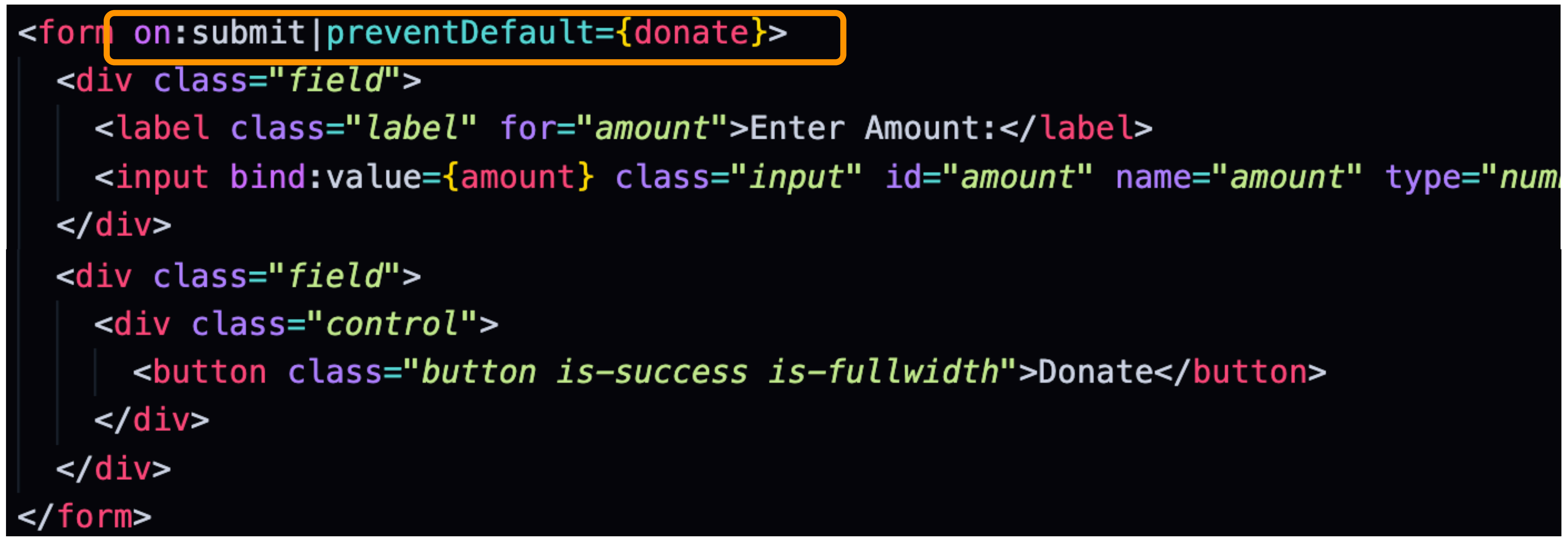
```
<form on:submit|preventDefault={donate}>
  <div class="field">
    <label class="label" for="amount">Enter Amount:</label>
    <input bind:value={amount} class="input" id="amount" name="amount" type="num
  </div>
  <div class="field">
    <div class="control">
      <button class="button is-success is-fullwidth">Donate</button>
    </div>
  </div>
</form>
```

- Switch to conventional HTTO POST.



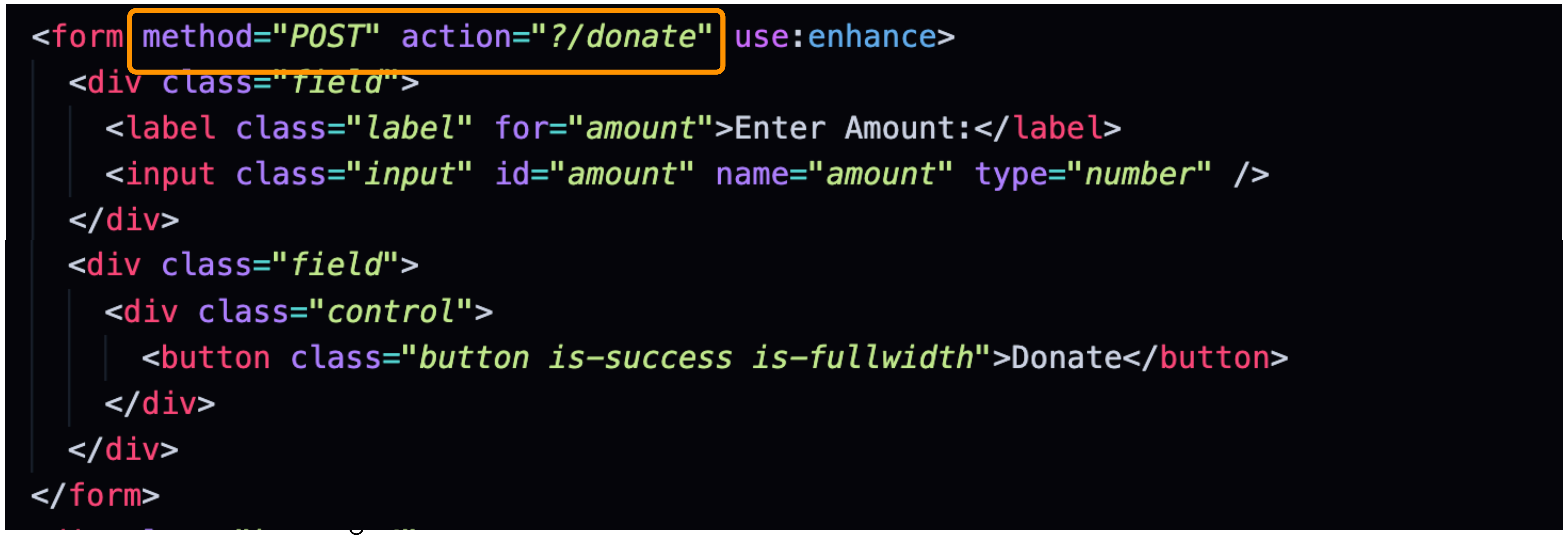
```
<form method="POST" action="?/donate" use:enhance>
  <div class="field">
    <label class="label" for="amount">Enter Amount:</label>
    <input class="input" id="amount" name="amount" type="number" />
  </div>
  <div class="field">
    <div class="control">
      <button class="button is-success is-fullwidth">Donate</button>
    </div>
  </div>
</form>
```

SSR Conversion: Server Side



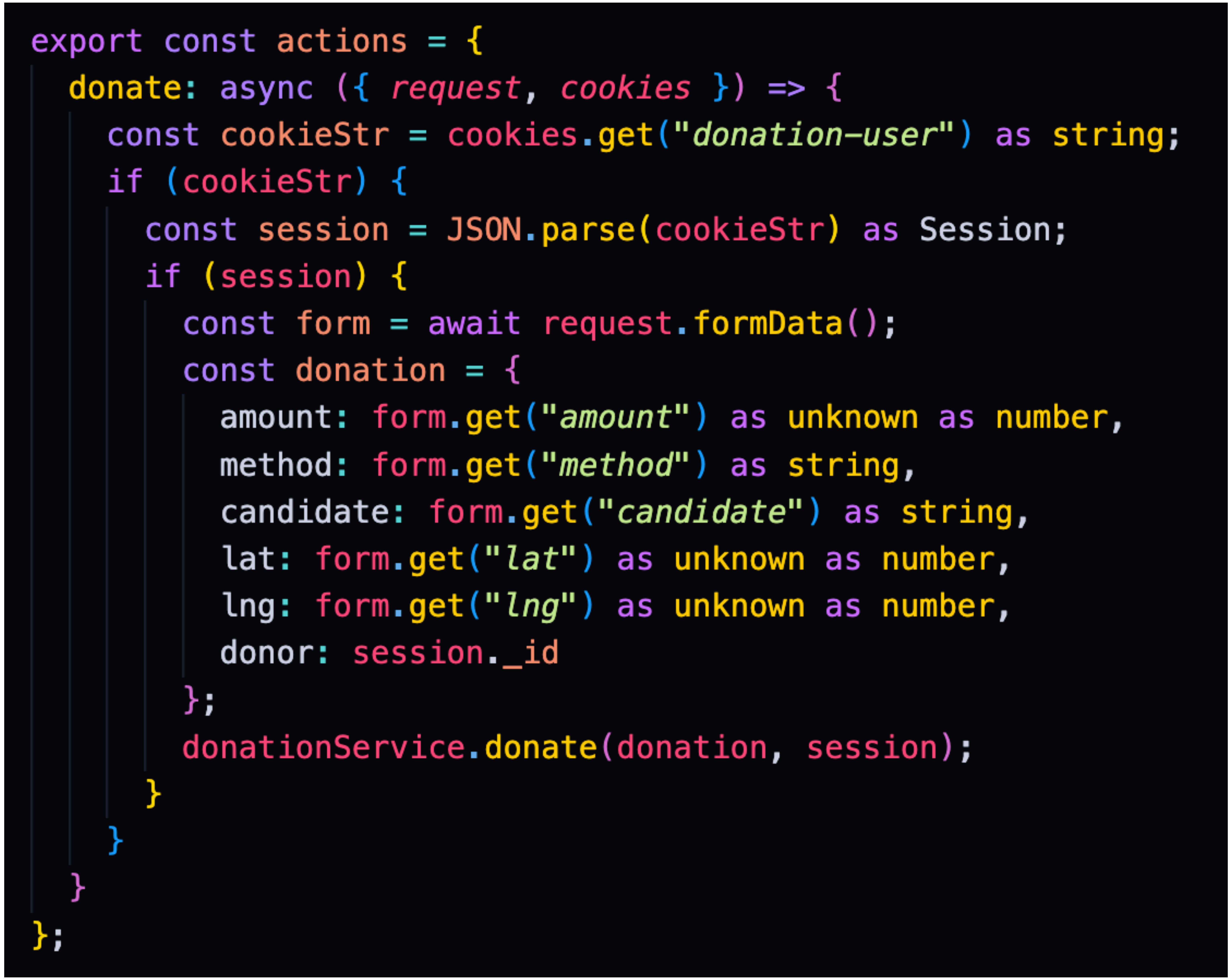
```
<form on:submit|preventDefault={donate}>
  <div class="field">
    <label class="label" for="amount">Enter Amount:</label>
    <input bind:value={amount} class="input" id="amount" name="amount" type="num
  </div>
  <div class="field">
    <div class="control">
      <button class="button is-success is-fullwidth">Donate</button>
    </div>
  </div>
</form>
```

- Send form data to server to be processed



```
<form method="POST" action="?/donate" use:enhance>
  <div class="field">
    <label class="label" for="amount">Enter Amount:</label>
    <input class="input" id="amount" name="amount" type="number" />
  </div>
  <div class="field">
    <div class="control">
      <button class="button is-success is-fullwidth">Donate</button>
    </div>
  </div>
</form>
```

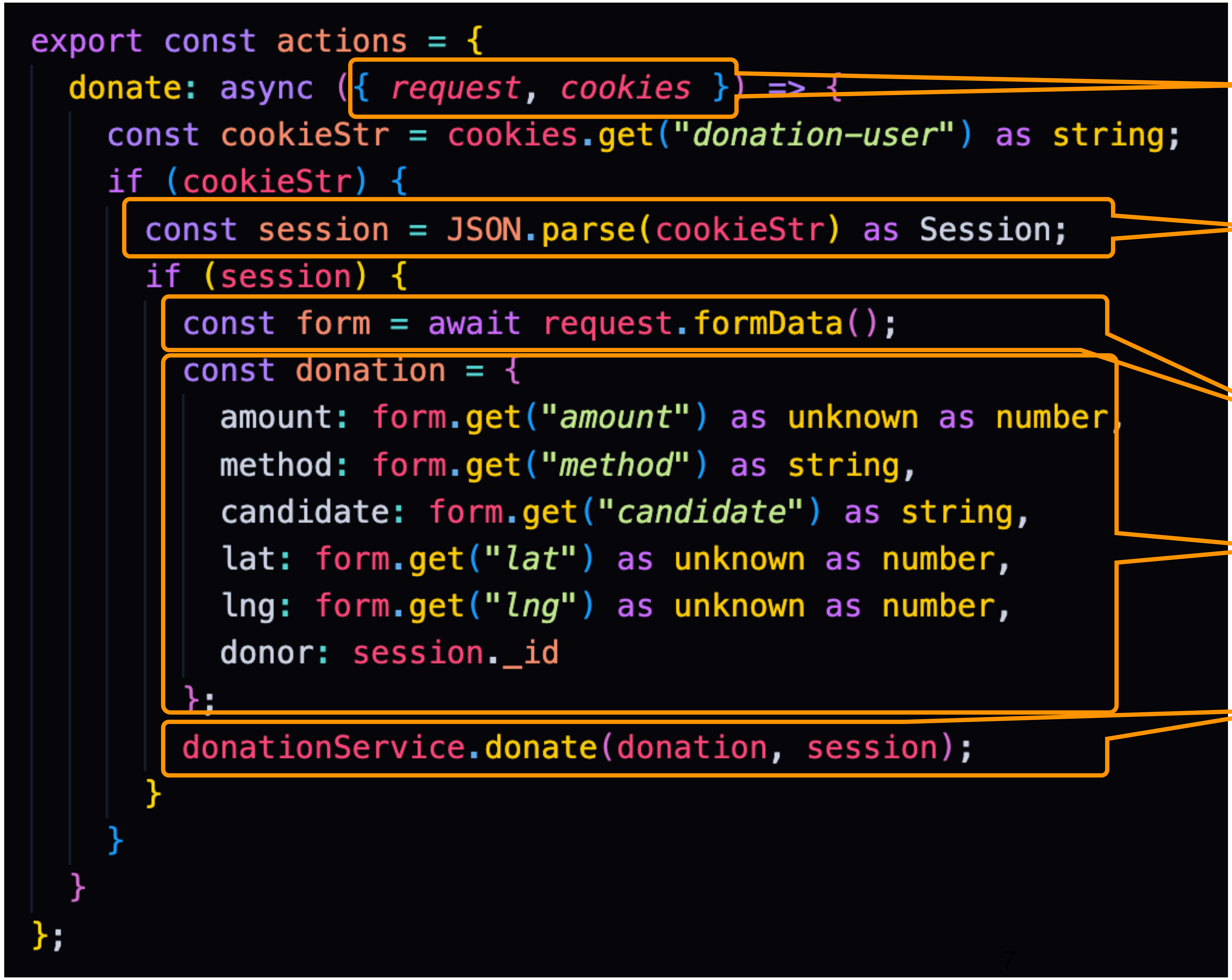
Donation Action



```
export const actions = {
  donate: async ({ request, cookies }) => {
    const cookieStr = cookies.get("donation-user") as string;
    if (cookieStr) {
      const session = JSON.parse(cookieStr) as Session;
      if (session) {
        const form = await request.formData();
        const donation = {
          amount: form.get("amount") as unknown as number,
          method: form.get("method") as string,
          candidate: form.get("candidate") as string,
          lat: form.get("lat") as unknown as number,
          lng: form.get("lng") as unknown as number,
          donor: session._id
        };
        donationService.donate(donation, session);
      }
    }
  }
};
```

- Request & Cookies passed as parameters
- Recover the session from the cookie
- Retrieve the form data
- Create a donation object
- Make the donation to the service

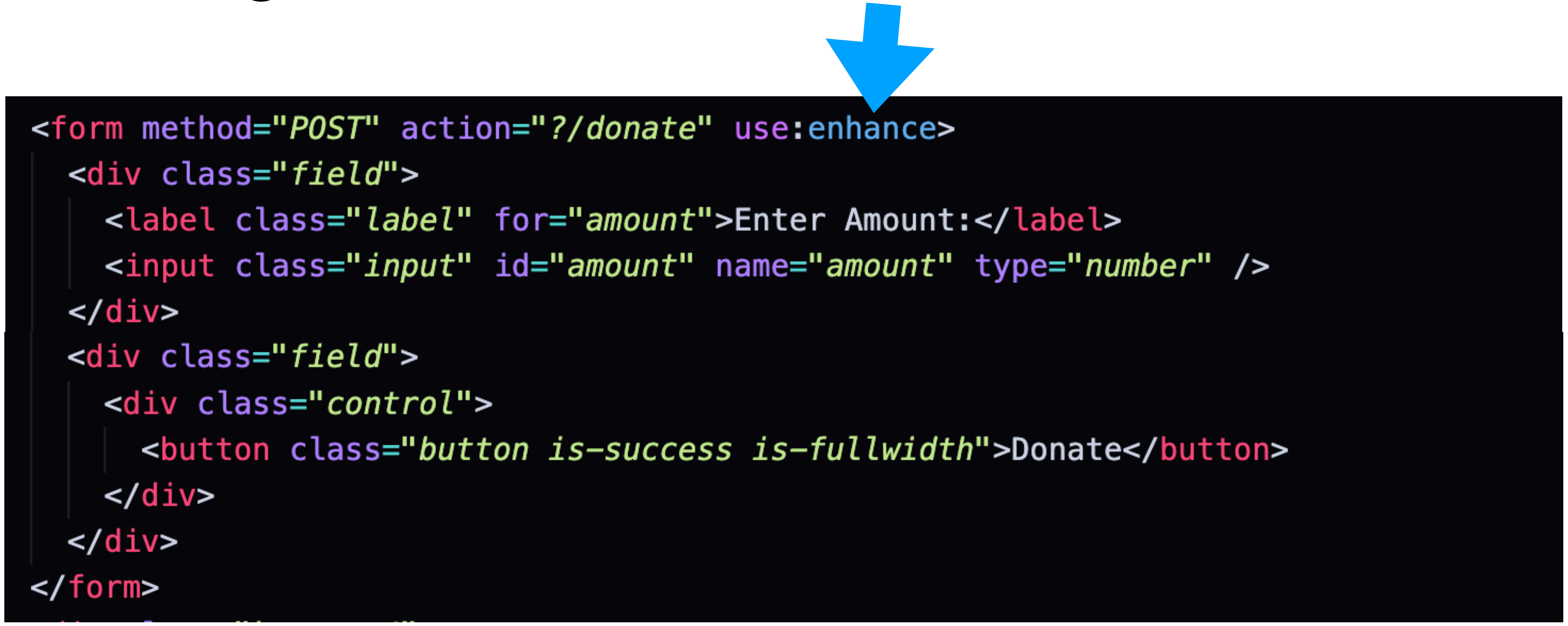
Donation Action



```
export const actions = {
  donate: async ({ request, cookies }) => {
    const cookieStr = cookies.get("donation-user") as string;
    if (cookieStr) {
      const session = JSON.parse(cookieStr) as Session;
      if (session) {
        const form = await request.formData();
        const donation = {
          amount: form.get("amount") as unknown as number,
          method: form.get("method") as string,
          candidate: form.get("candidate") as string,
          lat: form.get("lat") as unknown as number,
          lng: form.get("lng") as unknown as number,
          donor: session._id
        };
        donationService.donate(donation, session);
      }
    }
  }
};
```

- Request & Cookies passed as parameters
- Recover the session from the cookie
- Retrieve the form data
- Create a donation object
- Make the donation to the service

Progressive enhancement use:enhance



```
<form method="POST" action="?/donate" use:enhance>
  <div class="field">
    <label class="label" for="amount">Enter Amount:</label>
    <input class="input" id="amount" name="amount" type="number" />
  </div>
  <div class="field">
    <div class="control">
      <button class="button is-success is-fullwidth">Donate</button>
    </div>
  </div>
</form>
```

- use:enhance will emulate the browser-native behaviour, without the full-page reloads

- To the user the behaviour closely matches the responsiveness and interactivity of full client side form handling

https://kit.svelte.dev/docs/form-actions#progressive-enhancement

Forms in SSR

Form handling in +page.server.ts