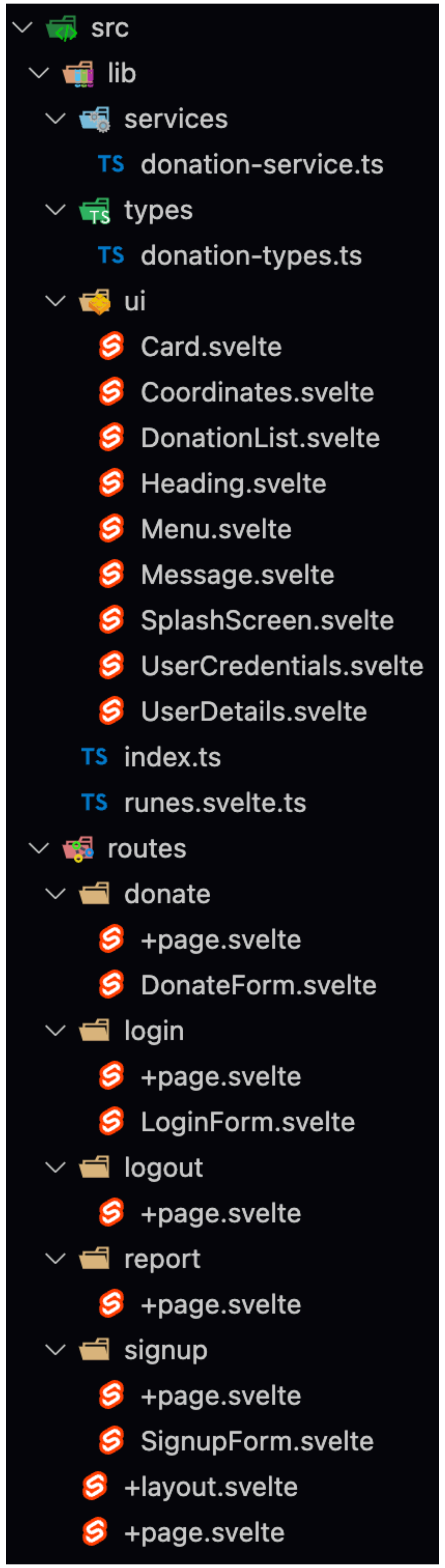
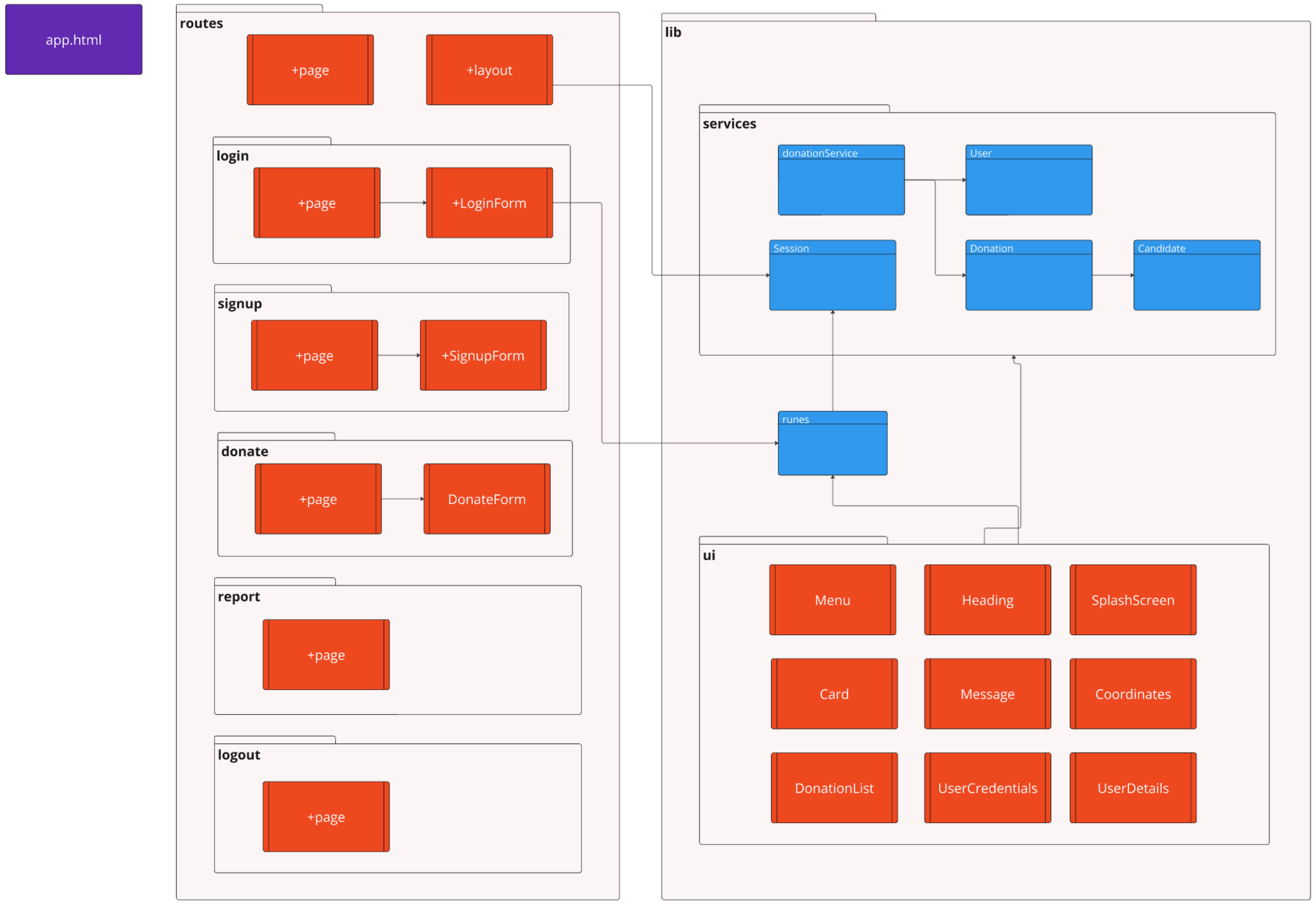
Donate in Svelte

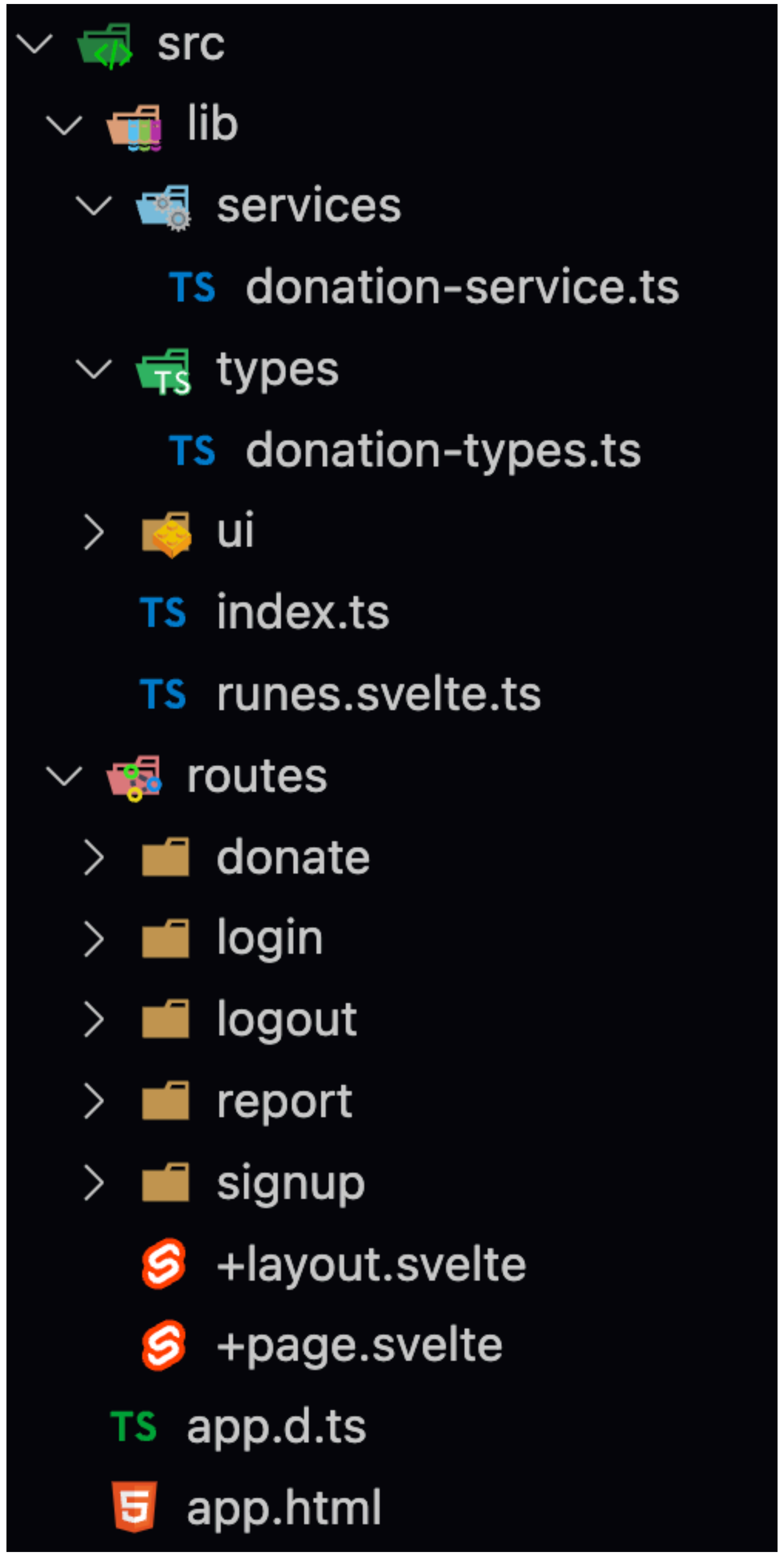
Implement the Donate & Report front end features



```
src
  lib
    services
      donation-service.ts
    types
      donation-types.ts
    ui
      Card.svelte
      Coordinates.svelte
      DonationList.svelte
      Heading.svelte
      Menu.svelte
      Message.svelte
      SplashScreen.svelte
      UserCredentials.svelte
      UserDetails.svelte
    index.ts
    runes.svelte.ts
  routes
    donate
      +page.svelte
      DonateForm.svelte
    login
      +page.svelte
      LoginForm.svelte
    logout
      +page.svelte
    report
      +page.svelte
    signup
      +page.svelte
      SignupForm.svelte
    +layout.svelte
    +page.svelte
```



Donation-types



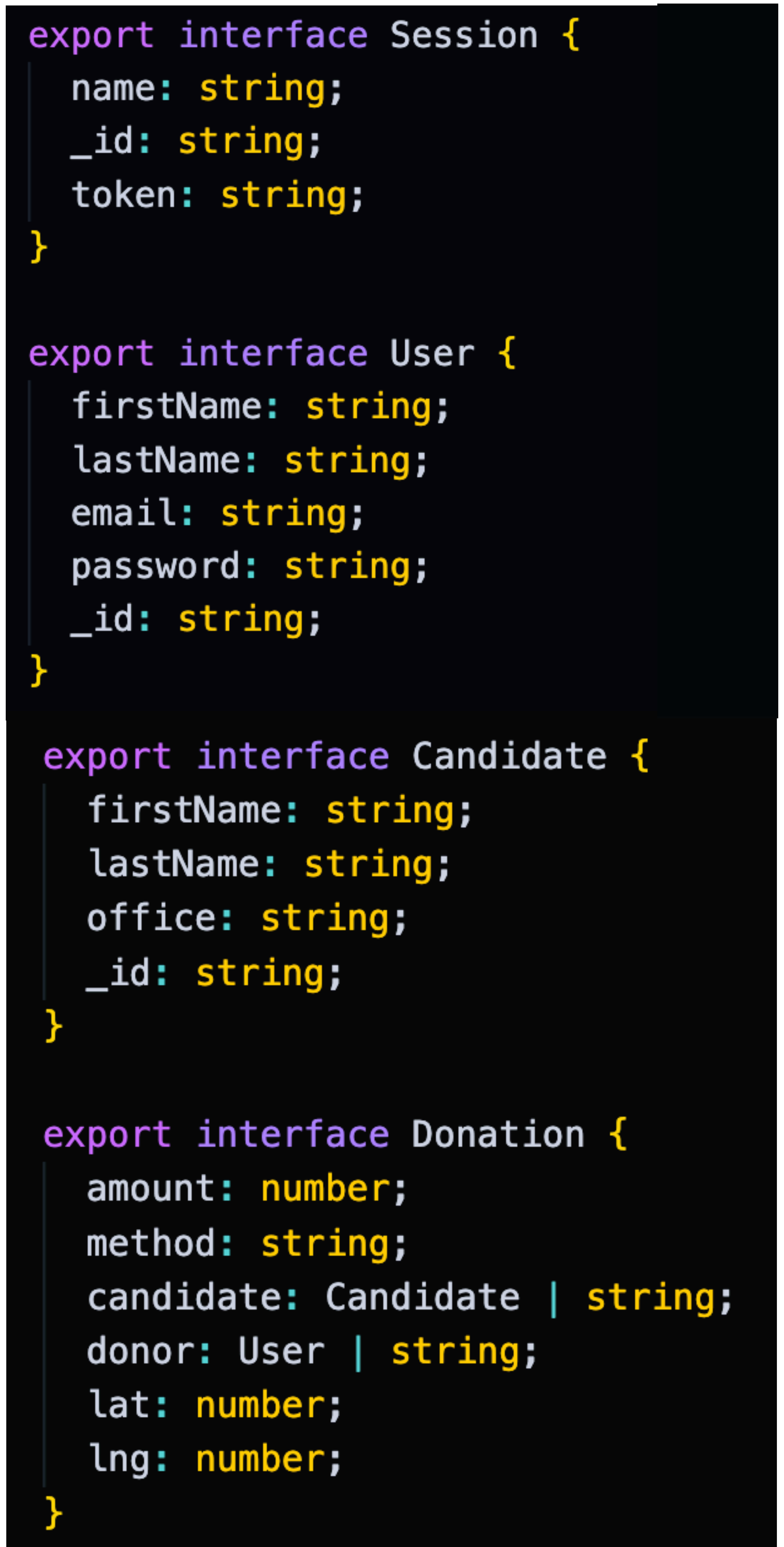
```
src
  lib
    services
      donation-service.ts
    types
      donation-types.ts
    ui
    index.ts
    runes.svelte.ts
  routes
    donate
    login
    logout
    report
    signup
    +layout.svelte
    +page.svelte
  app.d.ts
  app.html
```

Session: logged in user name, id + JWT token

User: user details

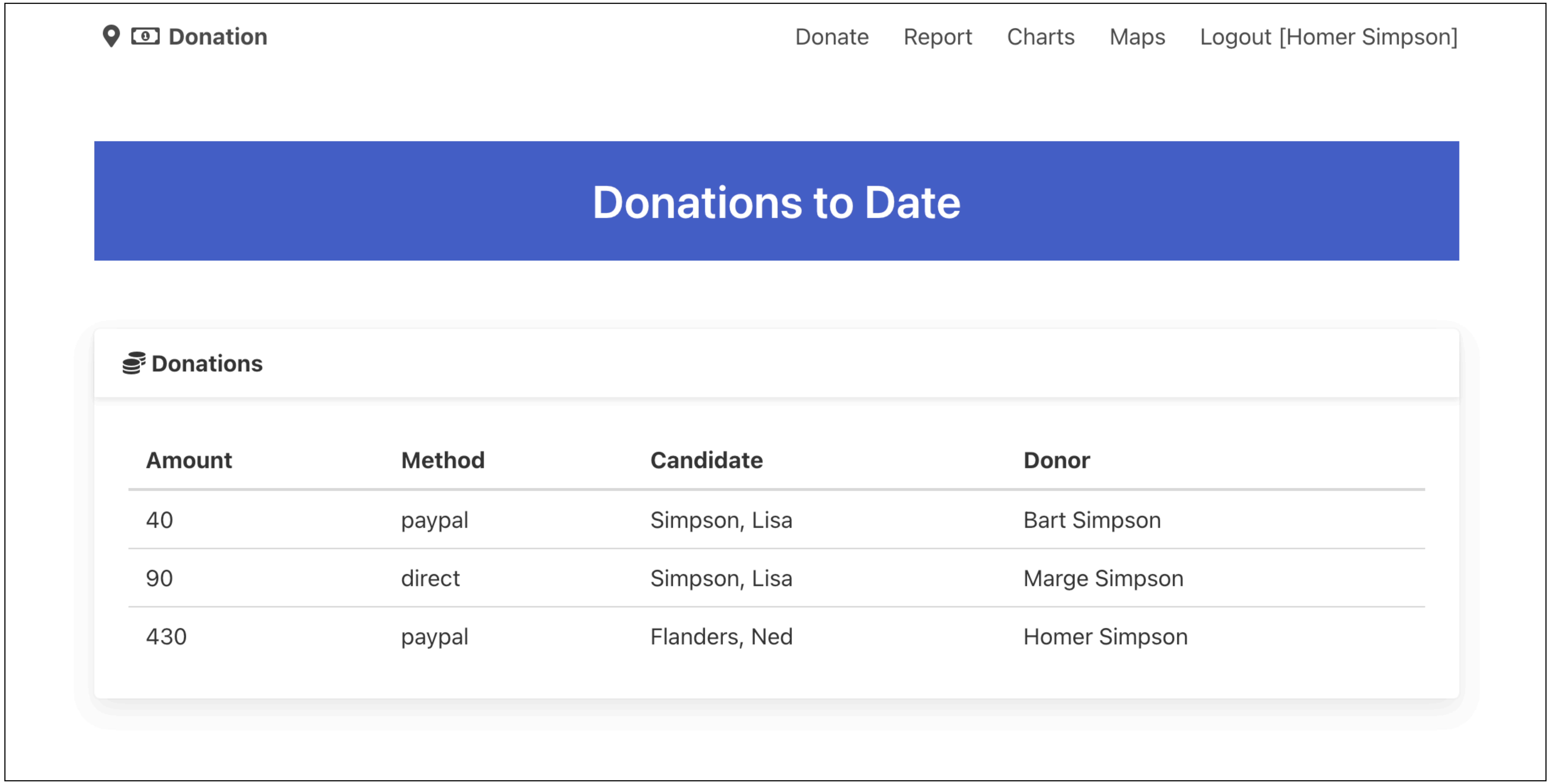
Candidate: Candidate details

Donation: the structure of a donation object



```
export interface Session {
  name: string;
  _id: string;
  token: string;
}

export interface User {
  firstName: string;
  lastName: string;
  email: string;
  password: string;
  _id: string;
}

export interface Candidate {
  firstName: string;
  lastName: string;
  office: string;
  _id: string;
}

export interface Donation {
  amount: number;
  method: string;
  candidate: Candidate | string;
  donor: User | string;
  lat: number;
  lng: number;
}
```

Report view



Donation

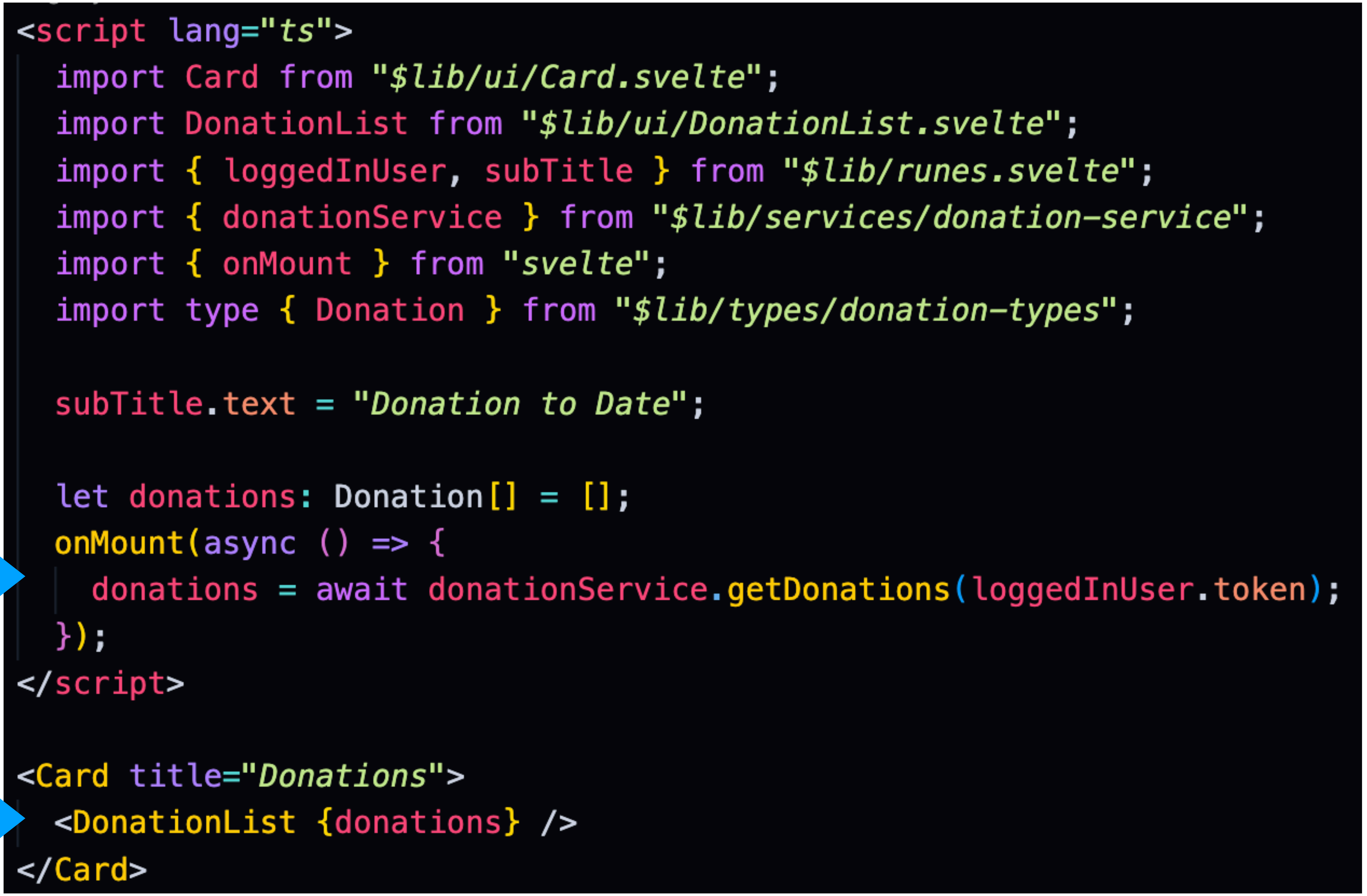
Donations to Date

Donations

Amount	Method	Candidate	Donor
40	paypal	Simpson, Lisa	Bart Simpson
90	direct	Simpson, Lisa	Marge Simpson
430	paypal	Flanders, Ned	Homer Simpson

For authenticated user, fetch all donations from Donation API

routes/report/+page.svelte

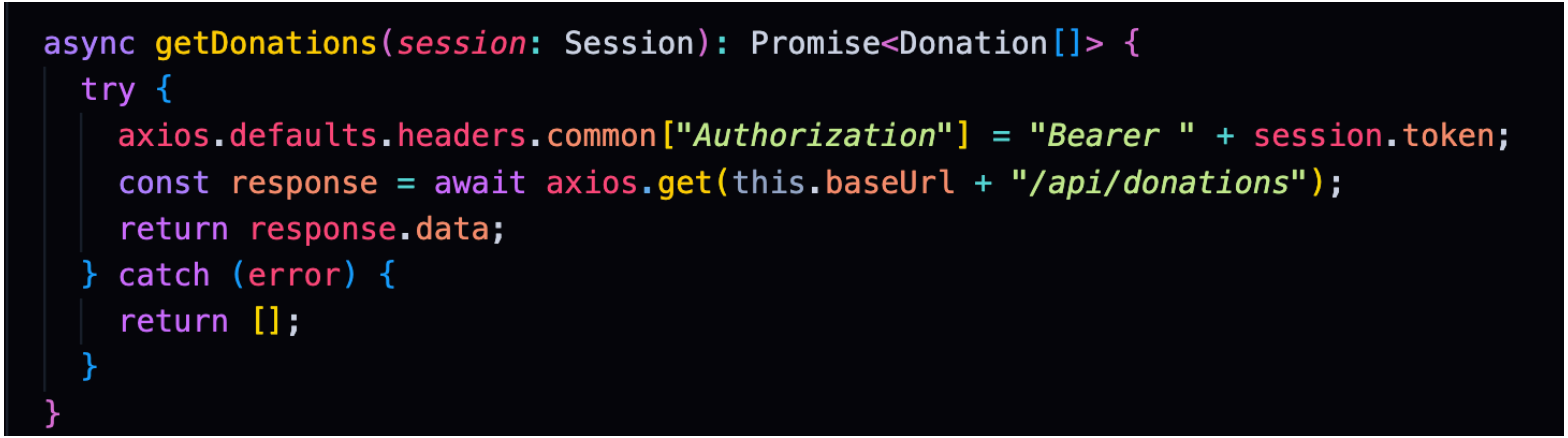


```
<script lang="ts">
  import Card from "$lib/ui/Card.svelte";
  import DonationList from "$lib/ui/DonationList.svelte";
  import { loggedInUser, subTitle } from "$lib/runes.svelte";
  import { donationService } from "$lib/services/donation-service";
  import { onMount } from "svelte";
  import type { Donation } from "$lib/types/donation-types";

  subTitle.text = "Donation to Date";

  let donations: Donation[] = [];
  onMount(async () => {
    donations = await donationService.getDonations(loggedInUser.token);
  });
</script>

<Card title="Donations">
  <DonationList {donations} />
</Card>
```

Retrieve donations from DonationList

Pass donations to DonationList component

donationService



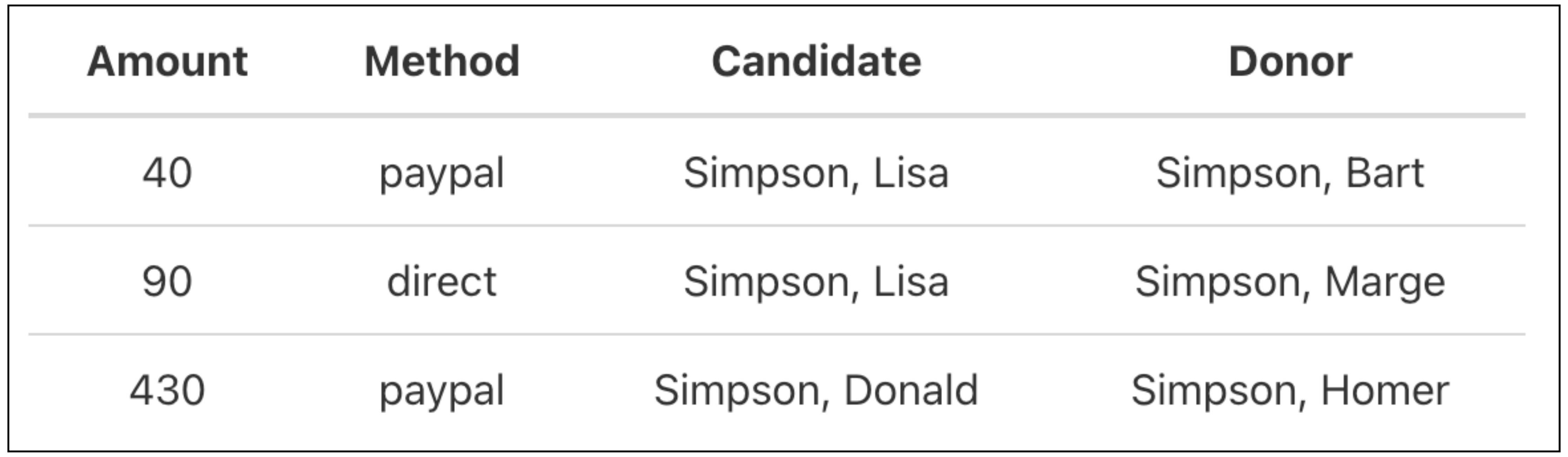
```
async getDonations(session: Session): Promise<Donation[]> {
  try {
    axios.defaults.headers.common["Authorization"] = "Bearer " + session.token;
    const response = await axios.get(this.baseUrl + "/api/donations");
    return response.data;
  } catch (error) {
    return [];
  }
}
```

- Retrieve all donations

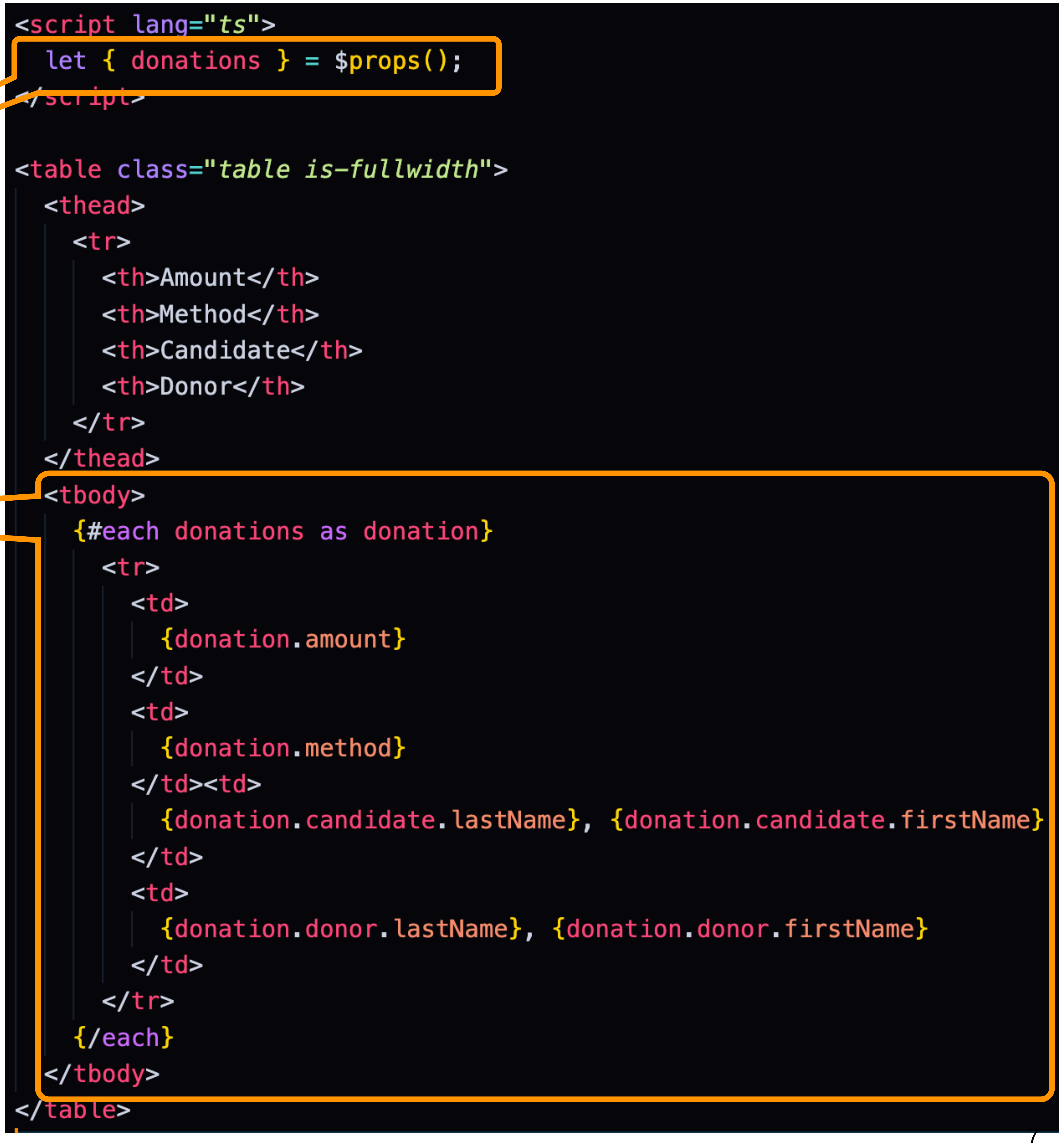
DonationList.svelte

- Donations array passed into the DonationList component

- Render donations in a table

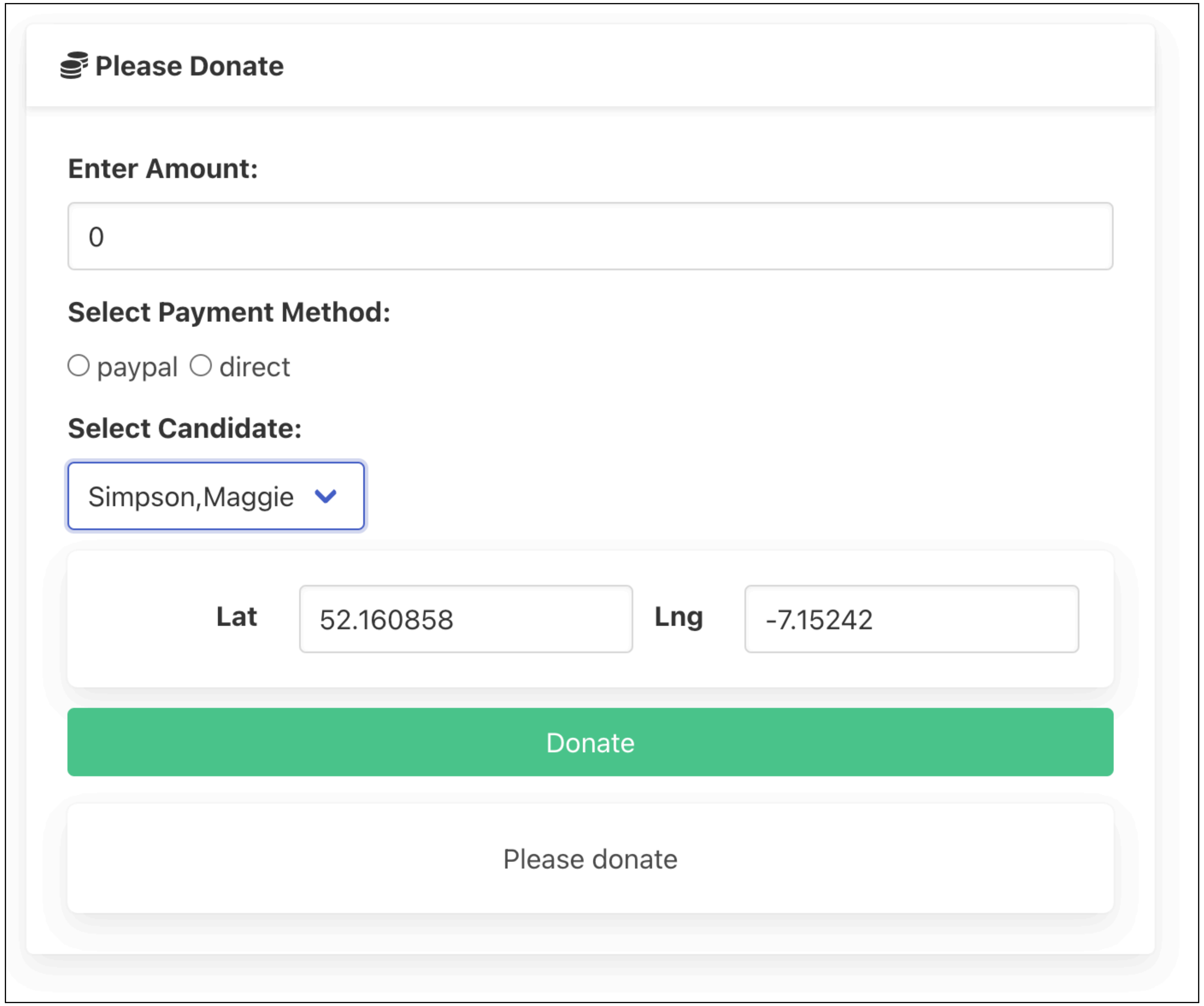


Amount	Method	Candidate	Donor
40	paypal	Simpson, Lisa	Simpson, Bart
90	direct	Simpson, Lisa	Simpson, Marge
430	paypal	Simpson, Donald	Simpson, Homer



```
<script lang="ts">
  let { donations } = $props();
</script>

<table class="table is-fullwidth">
  <thead>
    <tr>
      <th>Amount</th>
      <th>Method</th>
      <th>Candidate</th>
      <th>Donor</th>
    </tr>
  </thead>
  <tbody>
    {#each donations as donation}
      <tr>
        <td>
          {donation.amount}
        </td>
        <td>
          {donation.method}
        </td><td>
          {donation.candidate.lastName}, {donation.candidate.firstName}
        </td>
        <td>
          {donation.donor.lastName}, {donation.donor.firstName}
        </td>
      </tr>
    {/each}
  </tbody>
</table>
```

Donate



Please Donate

Enter Amount:

0

Select Payment Method:

○ paypal ○ direct

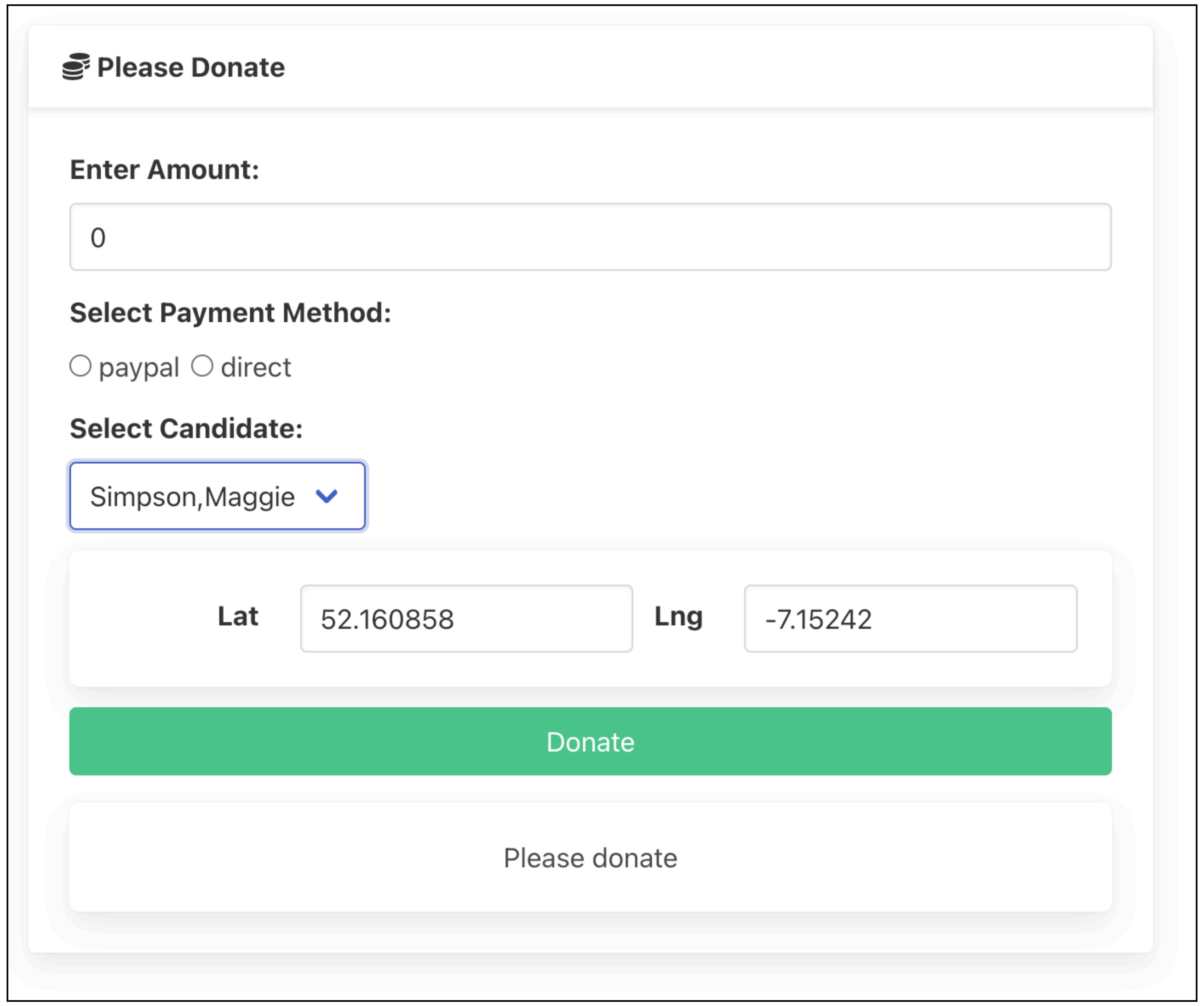
Select Candidate:

Simpson,Maggie

Lat	52.160858	**Lng**	-7.15242

Please donate

DonateForm.svelte



Please Donate

Enter Amount:

0

Select Payment Method:

○ paypal ○ direct

Select Candidate:

Simpson,Maggie

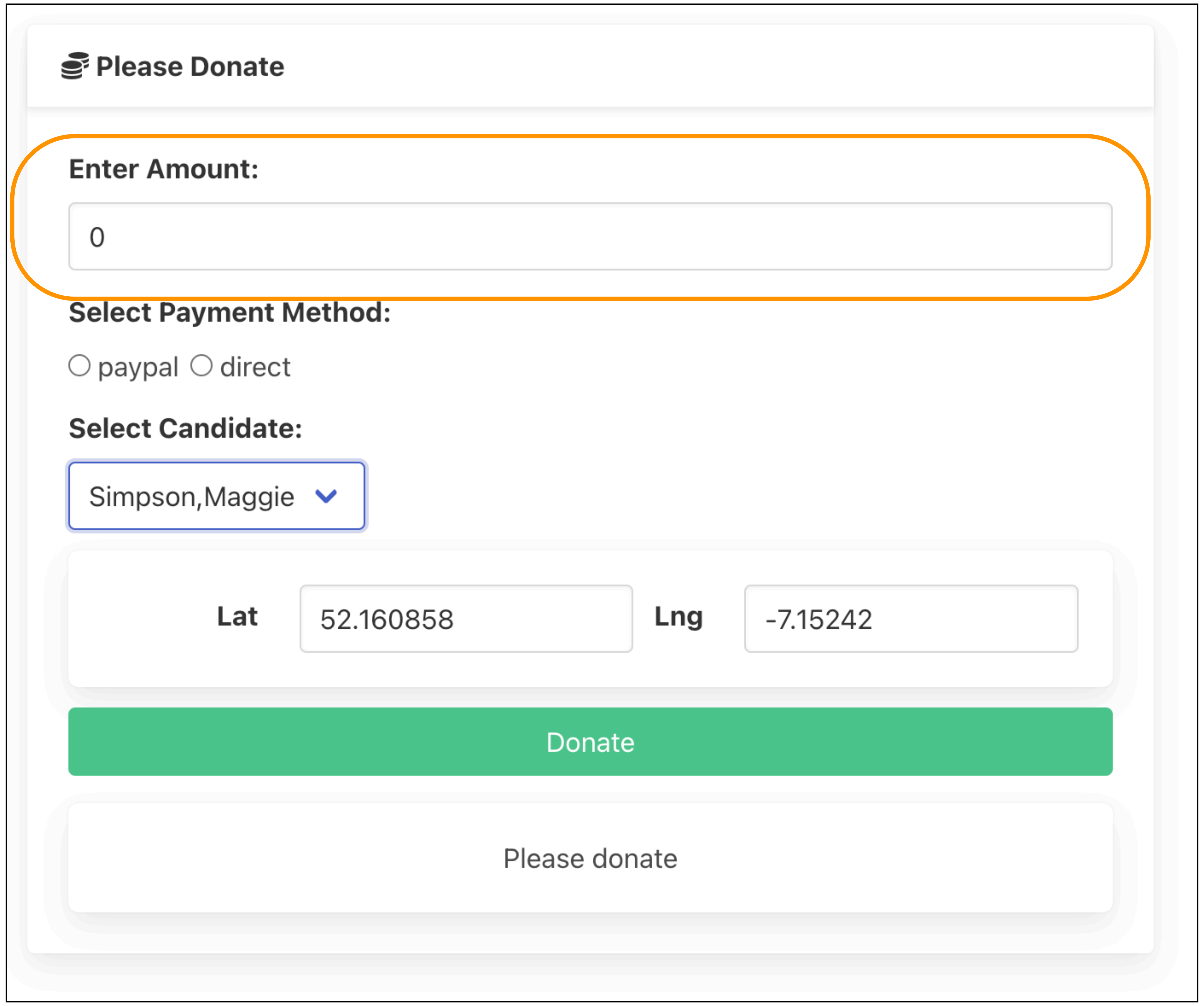
Lat 52.160858 Lng -7.15242

Donate

Please donate

```
<div>
  <div class="field">
    <label class="label" for="amount">Enter Amount:</label>
    <input bind:value={amount} class="input" id="amount" name="amount" type="number" />
  </div>
  <div class="field">
    <div class="control">
      <label class="label" for="amount">Select Payment Method:</label>
      {#each paymentMethods as method}
        <input bind:group={selectedMethod} class="radio" type="radio" value={method} /> {method}
      {/each}
    </div>
  </div>
  <div class="field">
    <label class="label" for="amount">Select Candidate:</label>
    <div class="select">
      <select bind:value={selectedCandidate}>
        {#each candidateList as candidate}
          <option value={candidate._id}>{candidate.lastName},{candidate.firstName}</option>
        {/each}
      </select>
    </div>
  </div>
  <div class="field">
    <div class="control">
      <button onclick={() => donate()} class="button">Donate</button>
    </div>
  </div>
</div>
<Coordinates bind:lat bind:lng />
<div class="box mt-4">
  <div class="content has-text-centered">
    {message}
  </div>
</div>
```

Input



Please Donate

Enter Amount:

0

Select Payment Method:

○ paypal ○ direct

Select Candidate:

Simpson,Maggie

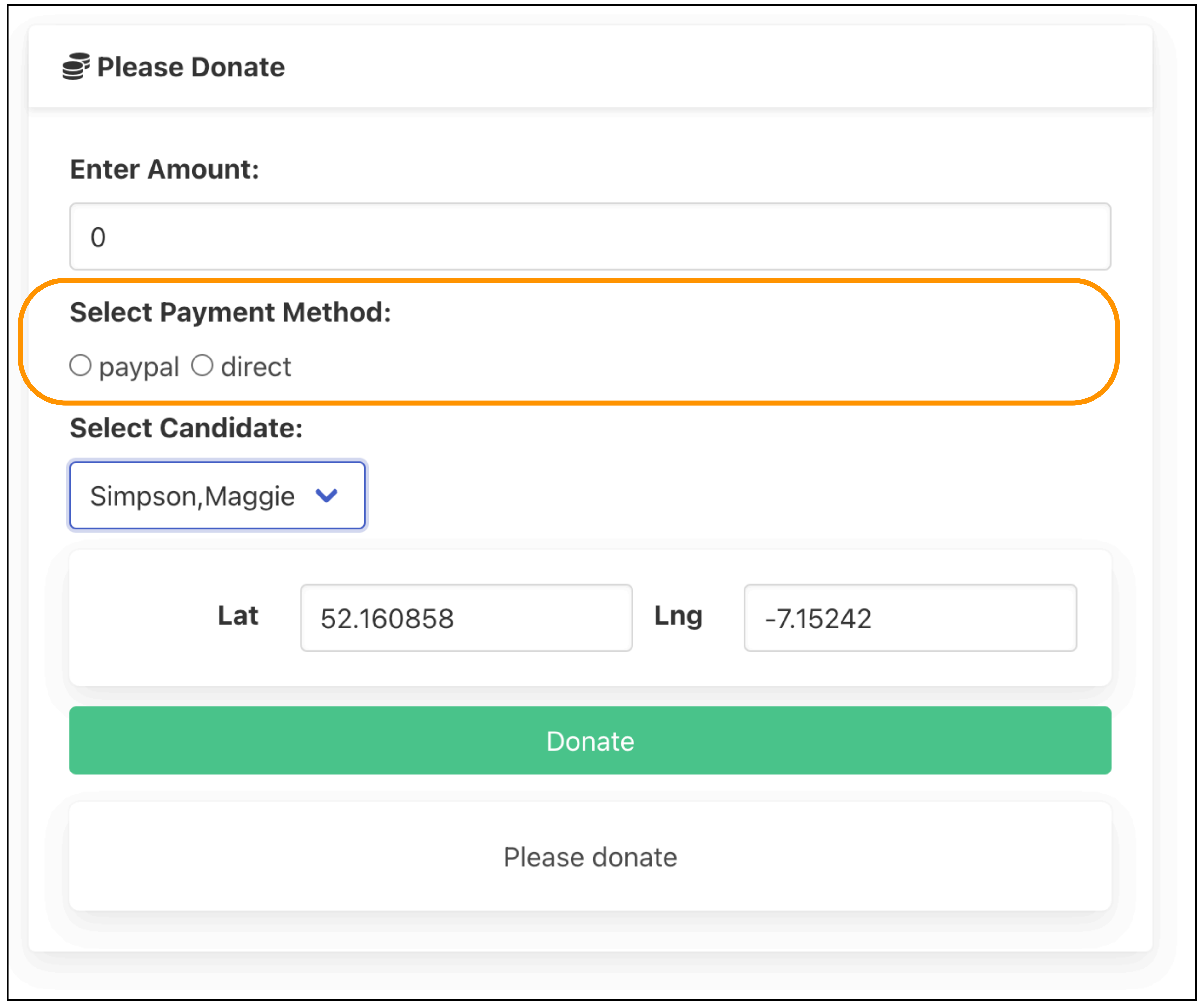
Lat 52.160858 Lng -7.15242

Please donate

```
let amount = $state(0);
```

```
<div class="field">
  <label class="label" for="amount">Enter Amount:</label>
  <input bind:value={amount} class="input" id="amount" name="amount" type="number" />
</div>
```

Radio


Please Donate

Enter Amount:

0

Select Payment Method:

○ paypal ○ direct

Select Candidate:

Simpson,Maggie

Lat 52.160858 Lng -7.15242

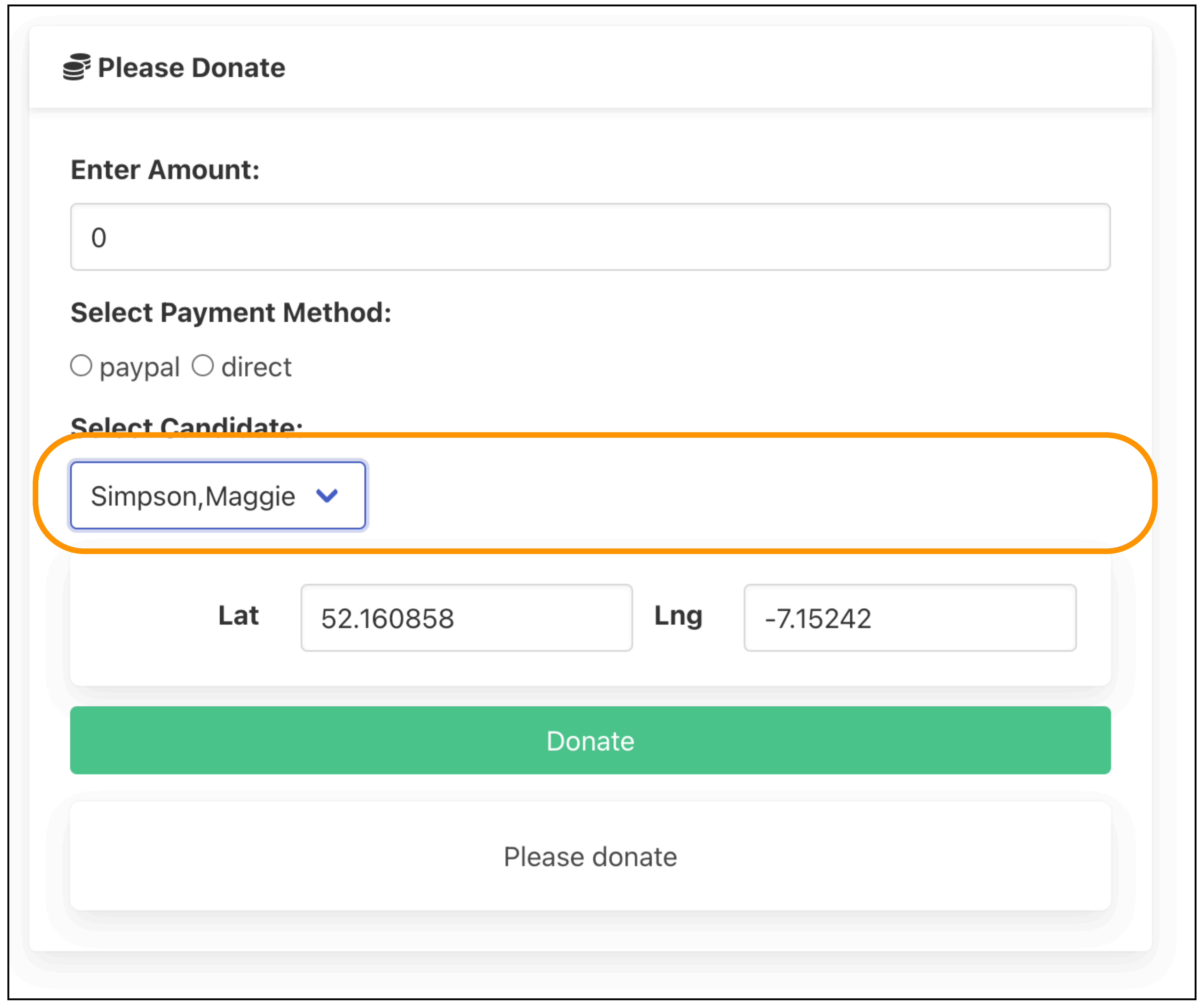
Donate

Please donate

```
let paymentMethods = ["paypal", "direct"];
let selectedMethod = $state("paypal");
```

```
<div class="control">
  <label class="label" for="amount">Select Payment Method:</label>
  {#each paymentMethods as method}
    <input bind:group={selectedMethod} class="radio" type="radio" value={method} /> {method}
  {/each}
</div>
```

Select / Dropdown


Please Donate

Enter Amount:

0

Select Payment Method:

○ paypal ○ direct

Select Candidate:

Simpson,Maggie

Lat 52.160858 Lng -7.15242

Donate

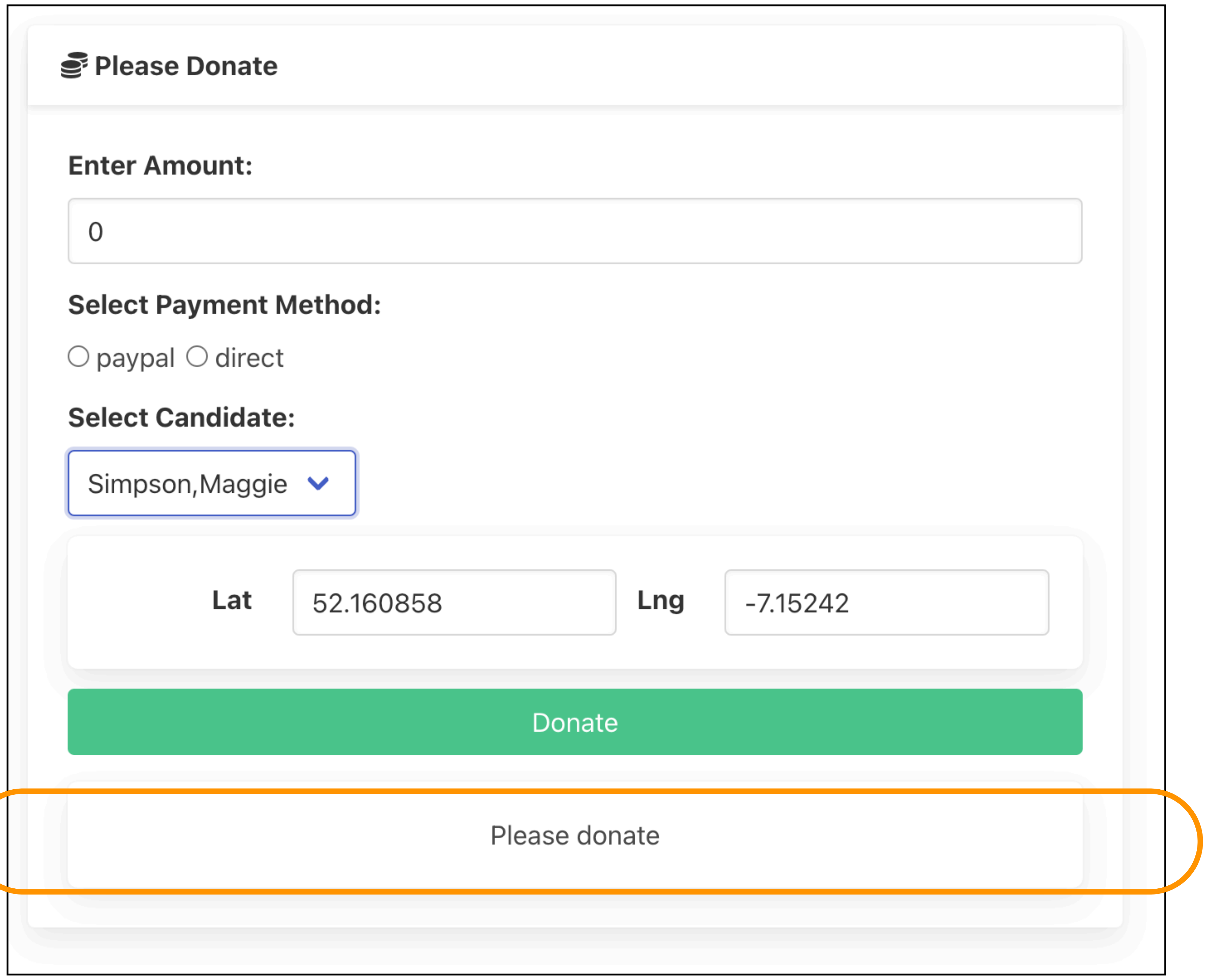
Please donate

```
export let candidateList: Candidate[] = [];
```

```
let selectedCandidate = "";
```

```
<div class="field">
  <label class="label" for="amount">Select Candidate:</label>
  <div class="select">
    <select bind:value={selectedCandidate}>
      {#each candidateList as candidate}
        <option value={candidate._id}>{candidate.lastName},{candidate.firstName}</option>
      {/each}
    </select>
  </div>
</div>
```

Text


Please Donate

Enter Amount:

0

Select Payment Method:

○ paypal ○ direct

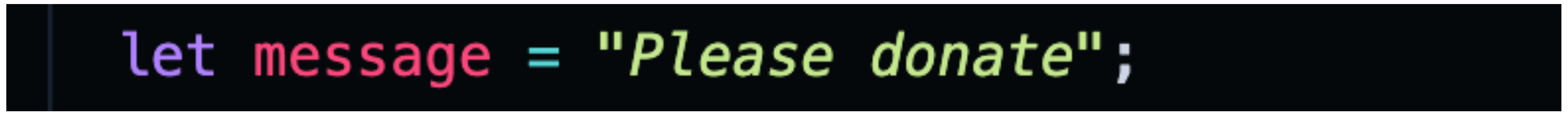
Select Candidate:

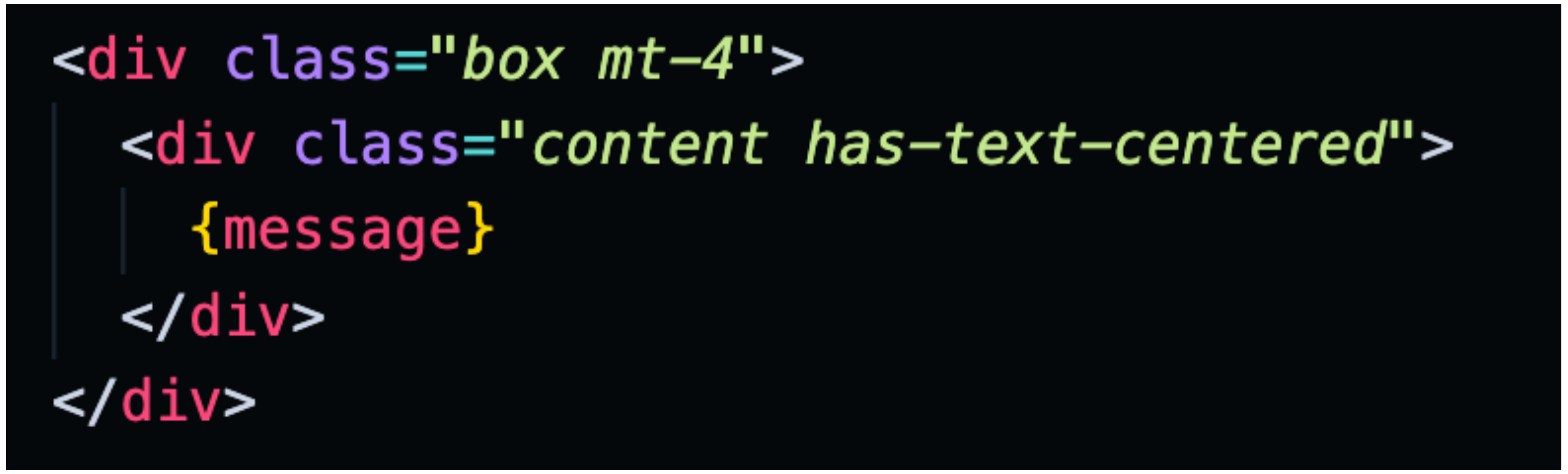
Simpson,Maggie

Lat 52.160858 Lng -7.15242

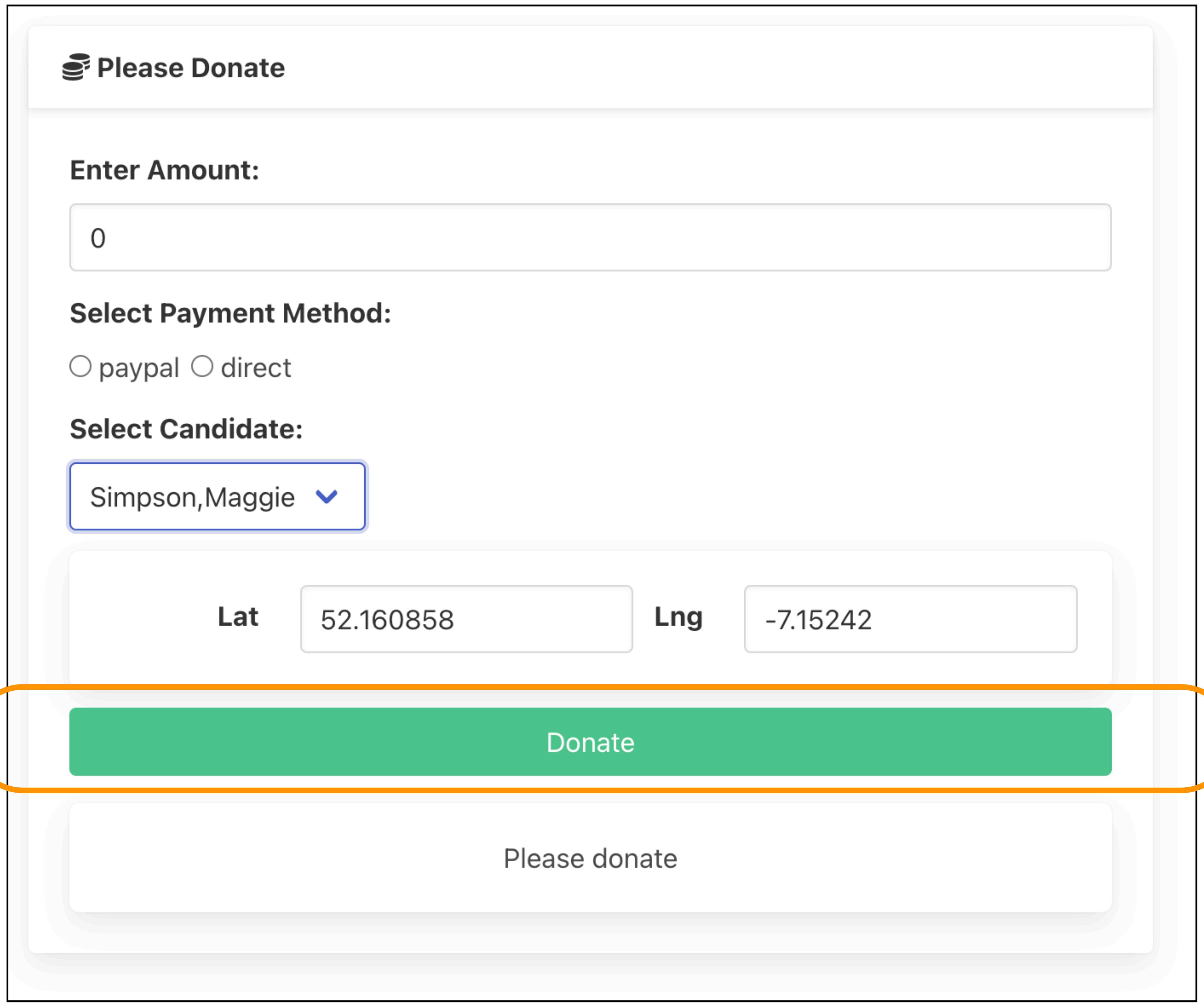
Donate

Please donate


```
let message = "Please donate";
```


```
<div class="box mt-4">
  <div class="content has-text-centered">
    {message}
  </div>
</div>
```

Button


Please Donate

Enter Amount:

0

Select Payment Method:

○ paypal ○ direct

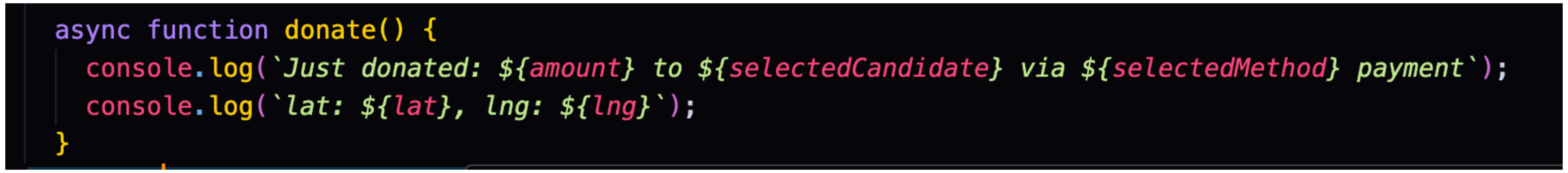
Select Candidate:

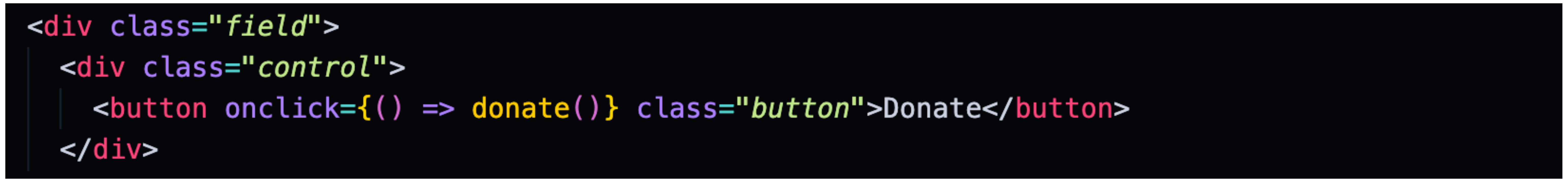
Simpson,Maggie

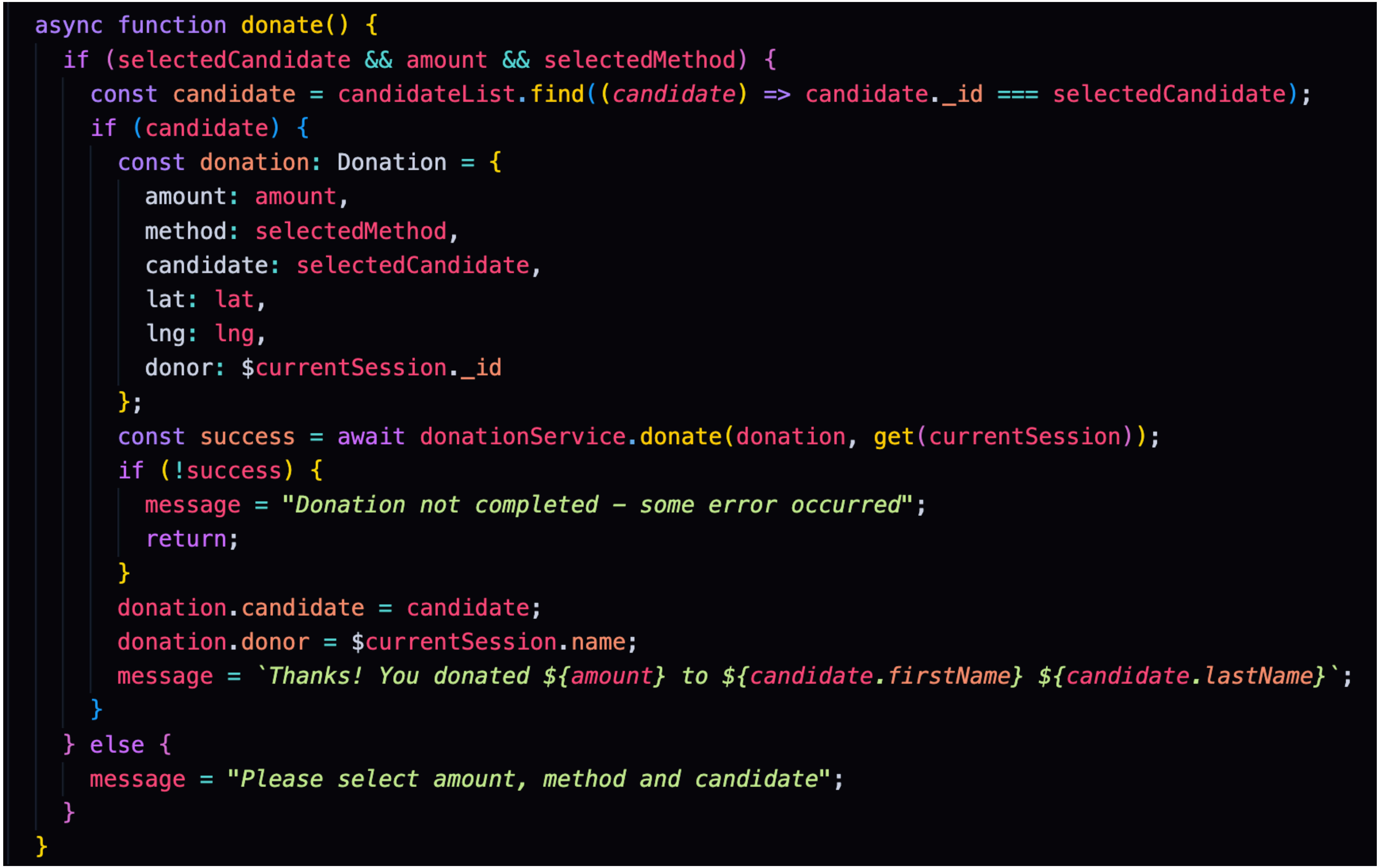
Lat 52.160858 Lng -7.15242

Donate

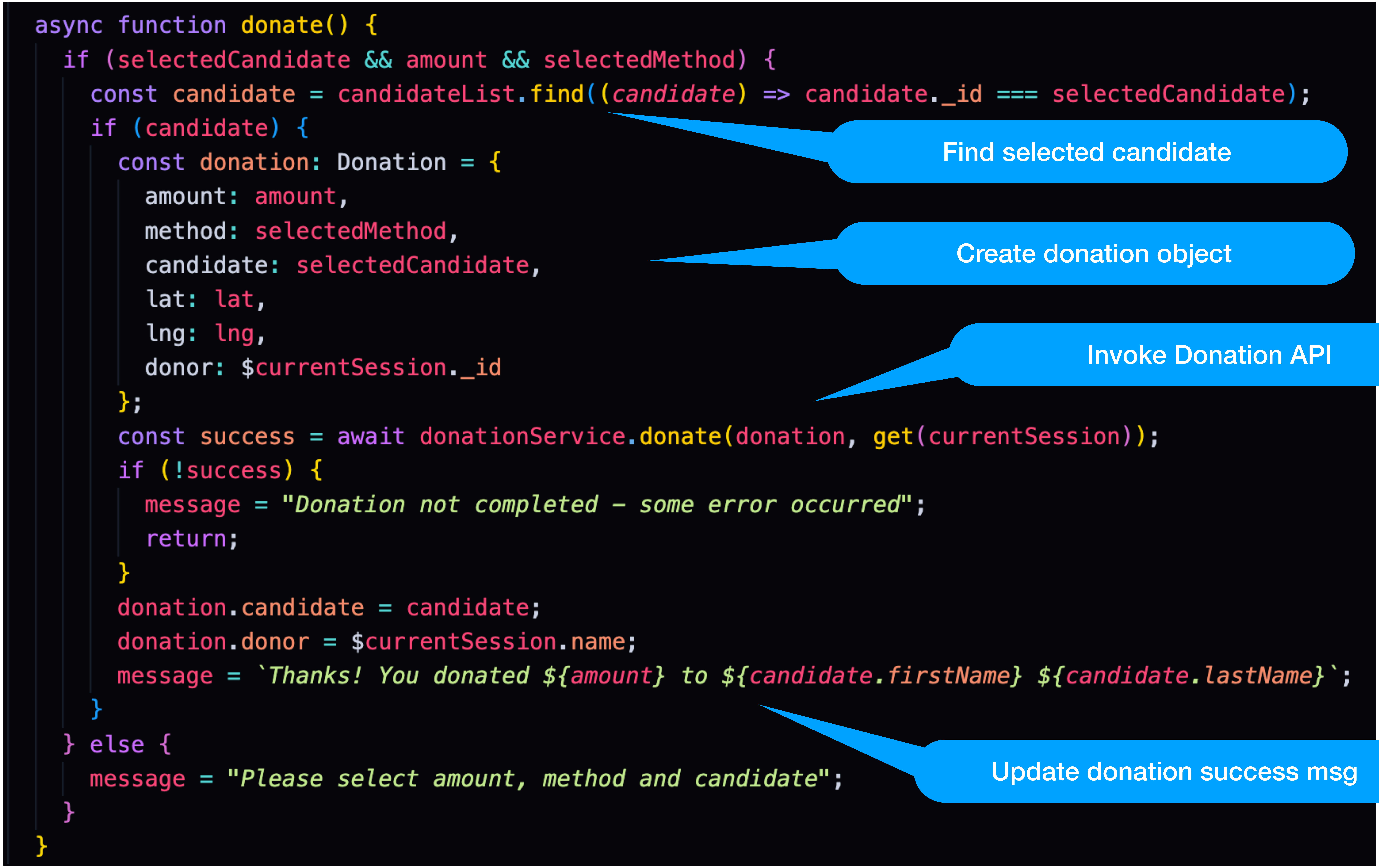
Please donate


```
async function donate() {
  console.log(`Just donated: ${amount} to ${selectedCandidate} via ${selectedMethod} payment`);
  console.log(`lat: ${lat}, lng: ${lng}`);
}
```


```
<div class="field">
  <div class="control">
    <button onclick={() => donate()} class="button">Donate</button>
  </div>
```



```
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;
      message = `Thanks! You donated ${amount} to ${candidate.firstName} ${candidate.lastName}`;
    }
  } else {
    message = "Please select amount, method and candidate";
  }
}
```



```
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;
      message = `Thanks! You donated ${amount} to ${candidate.firstName} ${candidate.lastName}`;
    }
  } else {
    message = "Please select amount, method and candidate";
  }
}
```

Donate in Svelte

Implement the Donate & Report front end features