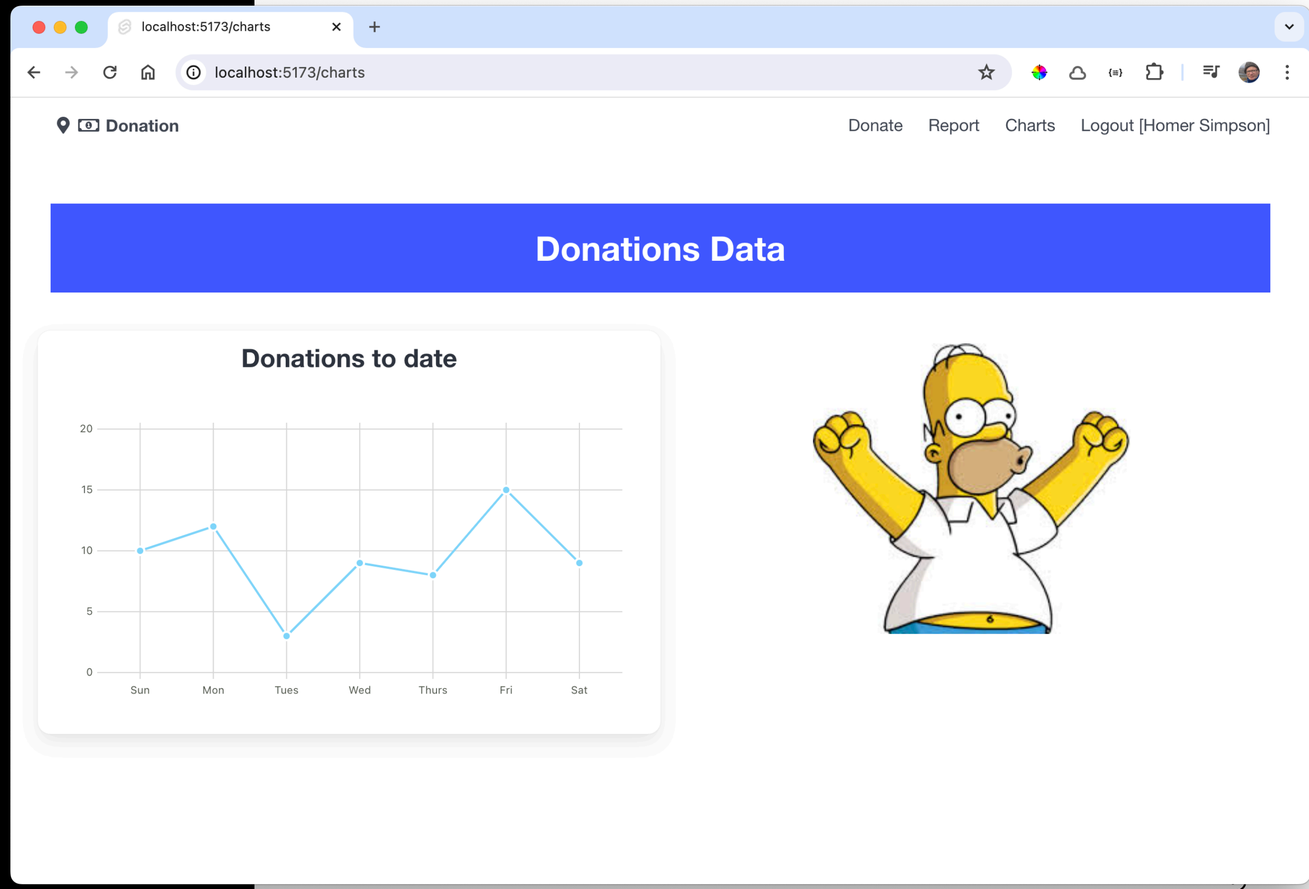
Charting Donations

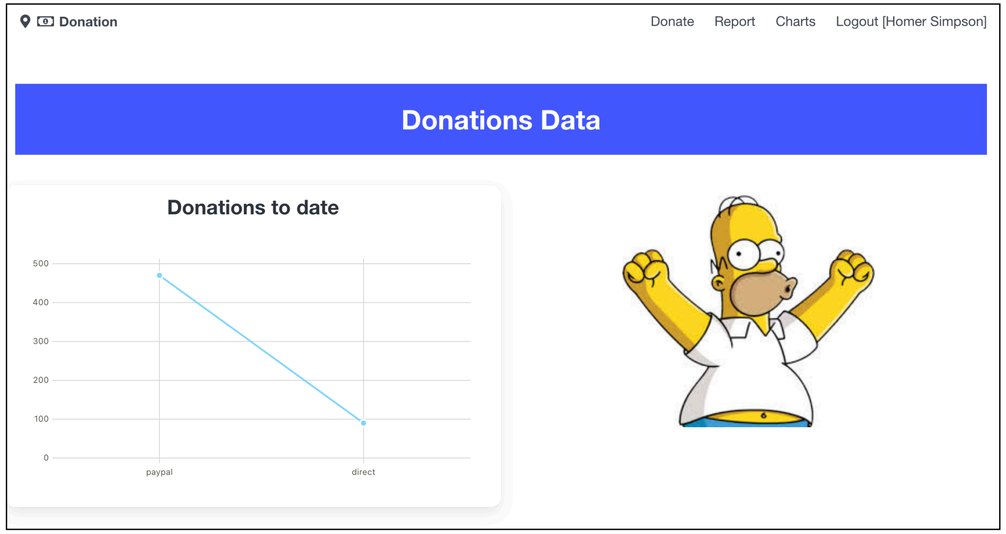
Plotting donations by candidate & payment method

```svelte
<script lang="ts">
  import { subTitle } from "$lib/runes.svelte";
  // @ts-ignore
  import Chart from "svelte-frappe-charts";

  subTitle.text = "Charts";
  const chartData = {
    labels: ["Sun", "Mon", "Tues", "Wed", "Thurs", "Fri", "Sat"],
    datasets: [
      {
        values: [10, 12, 3, 9, 8, 15, 9]
      }
    ]
  };
</script>

<div class="columns">
  <div class="column box has-text-centered">
    <h1 class="title is-4">Donations to date</h1>
    <Chart data={chartData} type="line" />
  </div>
  <div class="column has-text-centered">
    <img alt="Homer" src="/homer4.jpeg" width="300" />
  </div>
</div>
```

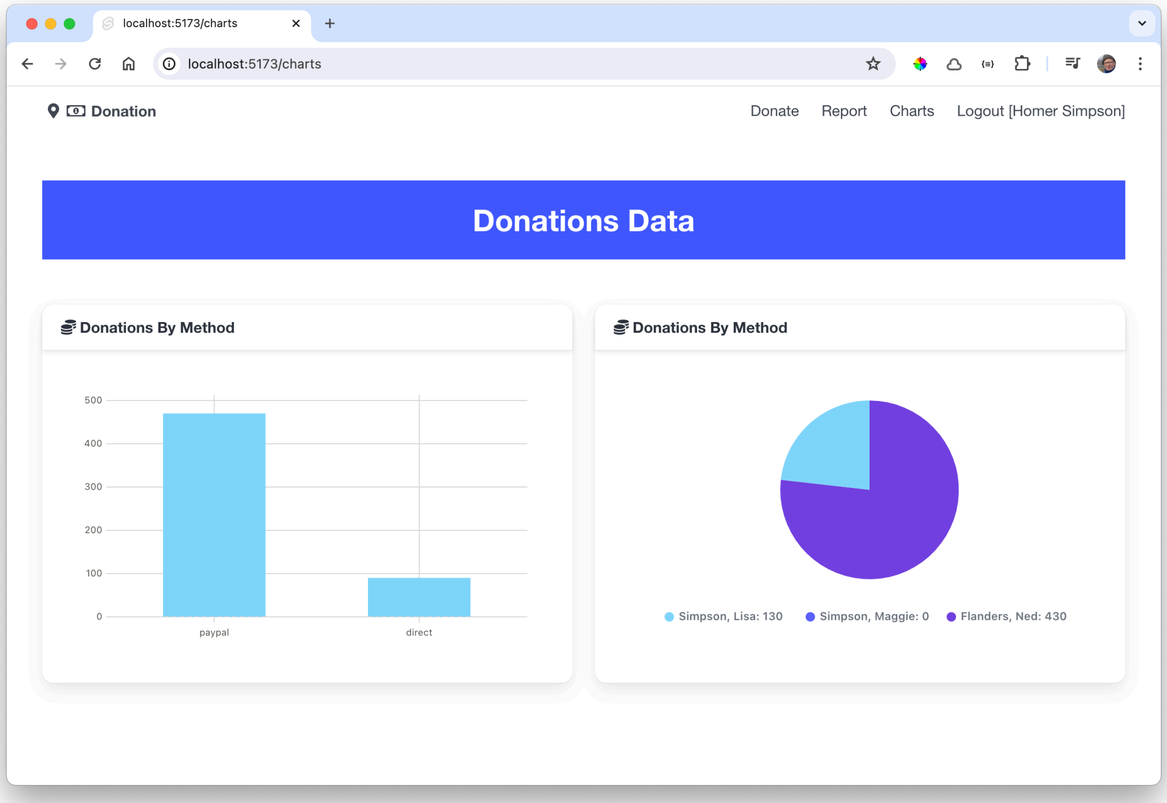
src/routes/charts/+page.svelte



- When the page loads
- Retrieve all donations
- Compute totals of ‘paypal’ & ‘direct’ payments



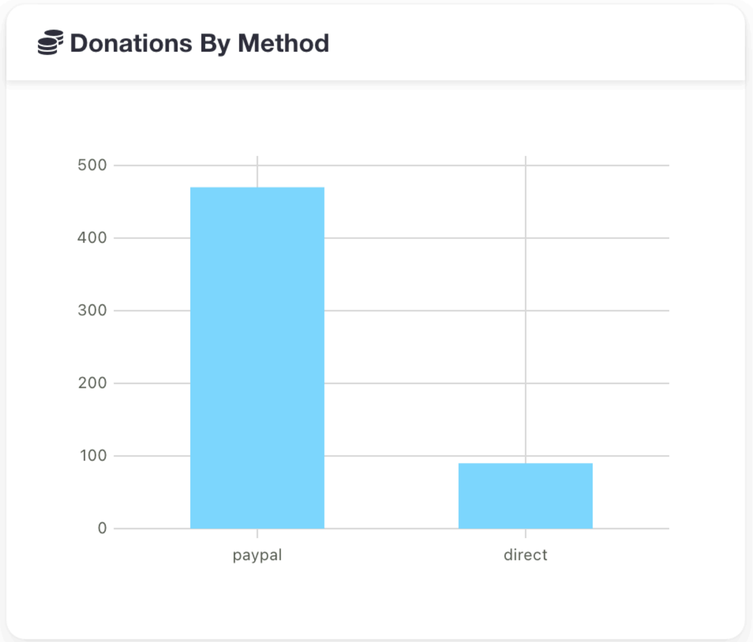

```svelte
<script lang="ts">
  import { subTitle } from "$lib/runes.svelte";
  import { onMount } from "svelte";
  import { currentDonations } from "$lib/runes.svelte";
  // @ts-ignore
  import Chart from "svelte-frappe-charts";

  subTitle.text = "Donations Data";
  const totalByMethod = {
    labels: ["paypal", "direct"],
    datasets: [
      {
        values: [0, 0]
      }
    ]
  };
  onMount(async () => {
    currentDonations.donations.forEach((donation) => {
      if (donation.method == "paypal") {
        totalByMethod.datasets[0].values[0] += donation.amount;
      } else if (donation.method == "direct") {
        totalByMethod.datasets[0].values[1] += donation.amount;
      }
    });
  });
</script>
```

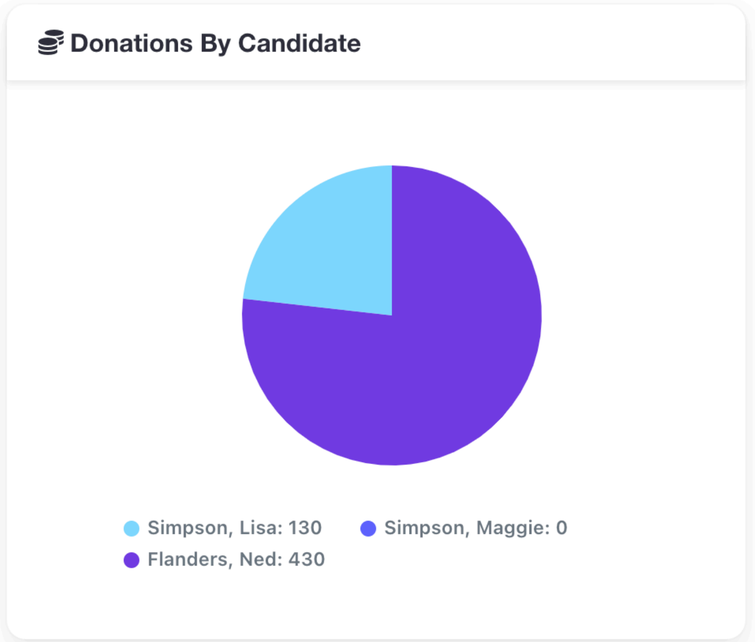



```
<div class="columns">
  <div class="column">
    <Card title="Donations By Method">
      <Chart data={totalByMethod} type="bar" />
    </Card>
  </div>
  <div class="column has-text-centered">
    <Card title="Donations By Method">
      <Chart data={totalByMethod} type="pie" />
    </Card>
  </div>
</div>
```

- Donations by payment methods: Pie + Bar
- 2 Chart components bound to the same data

- Donations by may payment method





- Donations by Candidate

```
const totalByMethod = {
  labels: ["paypal", "direct"],
  datasets: [
    {
      values: [0, 0]
    }
  ]
};
```

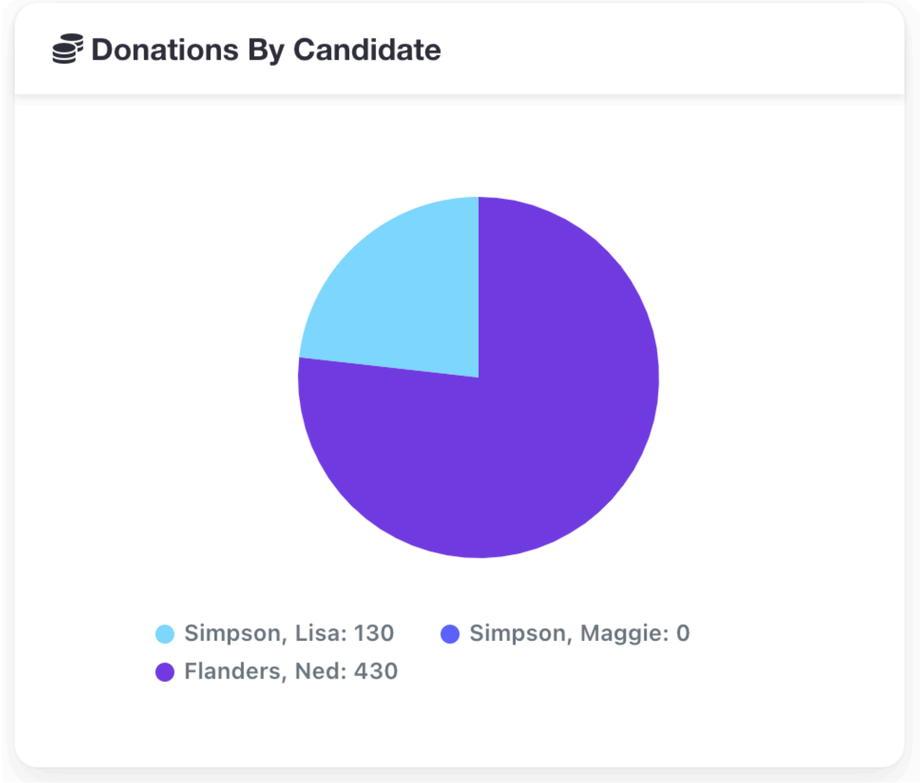
```
const donationsByCandidate = {
  labels: [],
  datasets: [
    {
      values: [0, 0]
    }
  ]
};
```

```
const donationsByCandidate = {
  labels: [],
  datasets: [
    {
      values: [0, 0]
    }
  ]
};
```


```
donationsByCandidate.labels = [];
currentCandidates.candidates.forEach((candidate, i) => {
  // @ts-ignore
  donationsByCandidate.labels.push(`${candidate.lastName}, ${candidate.firstName}`);
  donationsByCandidate.datasets[0].values.push(0);
});
```

- Retrieve the candidates and populate candidate labels

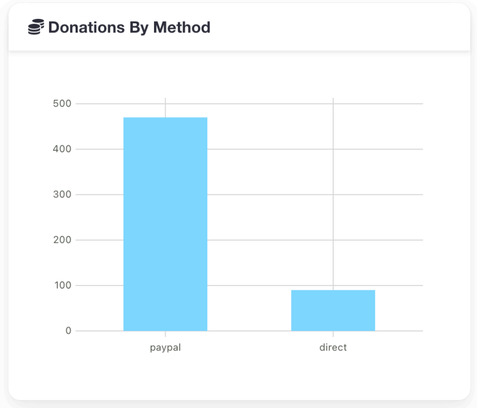
```
const donationsByCandidate = {
  labels: [],
  datasets: [
    {
      values: [0, 0]
    }
  ]
};
```



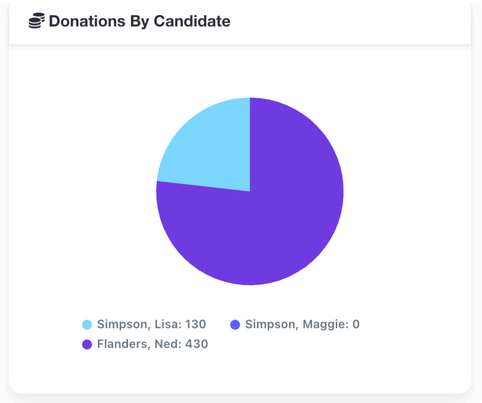

```
currentDonations.donations.forEach((donation) => {
  // @ts-ignore
  if (donation.candidate._id == candidate._id) {
    donationsByCandidate.datasets[0].values[i] += donation.amount;
  }
});
```

```
const totalByMethod = {
  labels: ["paypal", "direct"],
  datasets: [
    {
      values: [0, 0]
    }
  ]
};
```

```
const donationsByCandidate = {
  labels: [],
  datasets: [
    {
      values: [0, 0]
    }
  ]
};
```




```
<div class="columns">
  <div class="column">
    <Card title="Donations By Method">
      <Chart data={totalByMethod} type="bar" />
    </Card>
  </div>
  <div class="column has-text-centered">
    <Card title="Donations By Method">
      <Chart data={donationsByCandidate} type="pie" />
    </Card>
  </div>
</div>
```




```
currentDonations.donations.forEach((donation) => {
  if (donation.method == "paypal") {
    totalByMethod.datasets[0].values[0] += donation.amount;
  } else if (donation.method == "direct") {
    totalByMethod.datasets[0].values[1] += donation.amount;
  }
});
```

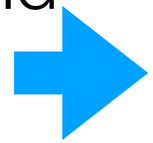
```
donationsByCandidate.labels = [];
currentCandidates.candidates.forEach((candidate, i) => {
  // @ts-ignore
  donationsByCandidate.labels.push(`${candidate.lastName}, ${candidate.firstName}`);
  donationsByCandidate.datasets[0].values.push(0);

  currentDonations.donations.forEach((donation) => {
    // @ts-ignore
    if (donation.candidate._id == candidate._id) {
      donationsByCandidate.datasets[0].values[i] += donation.amount;
    }
  });
});
```

donation-types.ts

```ts
export interface DataSet {
  labels: string[];
  datasets: [{ values: number[] }];
}
```

- Define type DataSet for donation chart data

- Define currentDataSets rune to hold latest donation analysis



runes.svelte.ts

```ts
export const curentDataSets = $state({
  donationsByMethod: {
    labels: ["paypal", "direct"],
    datasets: [
      {
        values: [0, 0]
      }
    ]
  },
  donationsByCandidate: {
    labels: [],
    datasets: [
      {
        values: [0, 0]
      }
    ]
  }
})
```

donation-utils.ts

```
import { curentDataSets } from "$lib/runes.svelte";
import type { Candidate, Donation } from "$lib/types/donation-types";
```

```
export function computeByMethod(donationList: Donation[]) {
  donationList.forEach((donation) => {
    if (donation.method == "paypal") {
      curentDataSets.donationsByMethod.datasets[0].values[0] += donation.amount;
    } else if (donation.method == "direct") {
      curentDataSets.donationsByMethod.datasets[0].values[1] += donation.amount;
    }
  });
}
```

- Refactor calculations in separate utility module

```
import { curentDataSets } from "$lib/runes.svelte";
import type { Candidate, Donation } from "$lib/types/donation-types";
```

donation-utils.ts

```
export function computeByCandidate(donationList: Donation[], candidates: Candidate[]) {
  curentDataSets.donationsByCandidate.labels = [];
  candidates.forEach((candidate) => {
    curentDataSets.donationsByCandidate.labels.push(
      // @ts-ignore
      `${candidate.lastName}, ${candidate.firstName}`
    );
    curentDataSets.donationsByCandidate.datasets[0].values.push(0);
  });

  candidates.forEach((candidate, i) => {
    donationList.forEach((donation) => {
      if (typeof donation.candidate !== "string") {
        if (donation.candidate._id == candidate._id) {
          curentDataSets.donationsByCandidate.datasets[0].values[i] += donation.amount;
        }
      }
    });
  });
}
```

donation-service.ts

```
async refreshDonationInfo() {
  if (loggedInUser.token) {
    currentDonations.donations = await this.getDonations(loggedInUser.token);
    currentCandidates.candidates = await this.getCandidates(loggedInUser.token);
    computeByMethod(currentDonations.donations);
    computeByCandidate(currentDonations.donations, currentCandidates.candidates)
  }
},
```

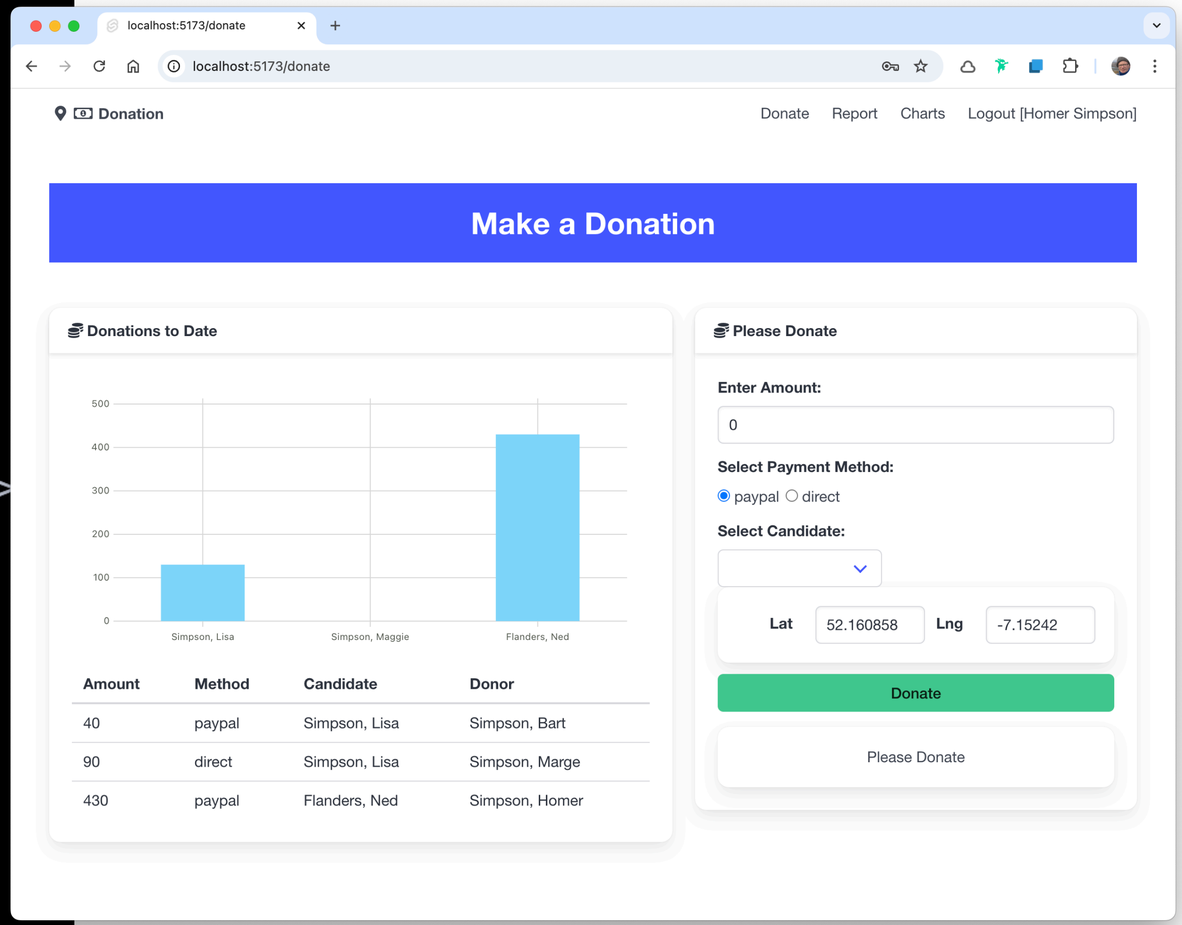
runes.svelte.ts

```
export const currentDonations = $state({
  donations: [] as Donation[],
});
export const currentCandidates = $state({ candidates: [] as Candidate[] });

export const curentDataSets = $state({
  donationsByMethod: {
    labels: ["paypal", "direct"],
    datasets: [
      {
        values: [0, 0]
      }
    ]
  },
  donationsByCandidate: {
    labels: [],
    datasets: [
      {
        values: [0, 0]
      }
    ]
  }
})
```

- When invoked, this method will fetch latest conations + candidate lists
- It will then calculate latest donation analytic information
- As all this information is store in runes, any components displaying this data will be automatically refreshed

Display Chart on Donate Page

```svelte
<script lang="ts">
  import { curentDataSets, subTitle } from "$lib/runes.svelte";
  import Card from "$lib/ui/Card.svelte";
  import DonateForm from "./DonateForm.svelte";
  // @ts-ignore
  import Chart from "svelte-frappe-charts";
  import DonationList from "$lib/ui/DonationList.svelte";

  subTitle.text = "Make a Donation";
</script>

<div class="columns">
  <div class="column">
    <Card title="Donations to Date">
      <Chart data={curentDataSets.donationsByCandidate} type="bar" />
      <DonationList />
    </Card>
  </div>
  <div class="column">
    <Card title="Please Donate">
      <DonateForm />
    </Card>
  </div>
</div>
```



- Simply import and place the Chart component

- Data will be automatically refreshed as donations are made

Charting Donations

Plotting donations by candidate & payment method