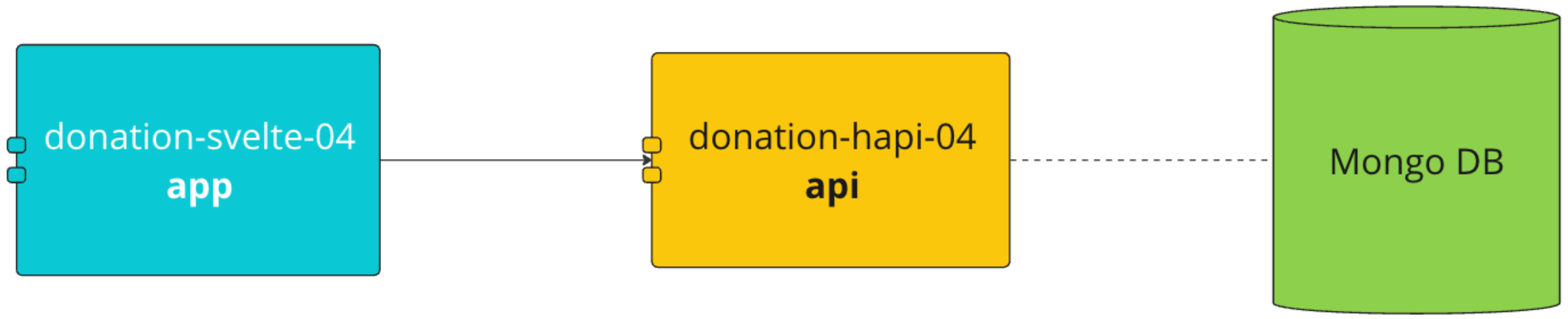
Svelte Authentication

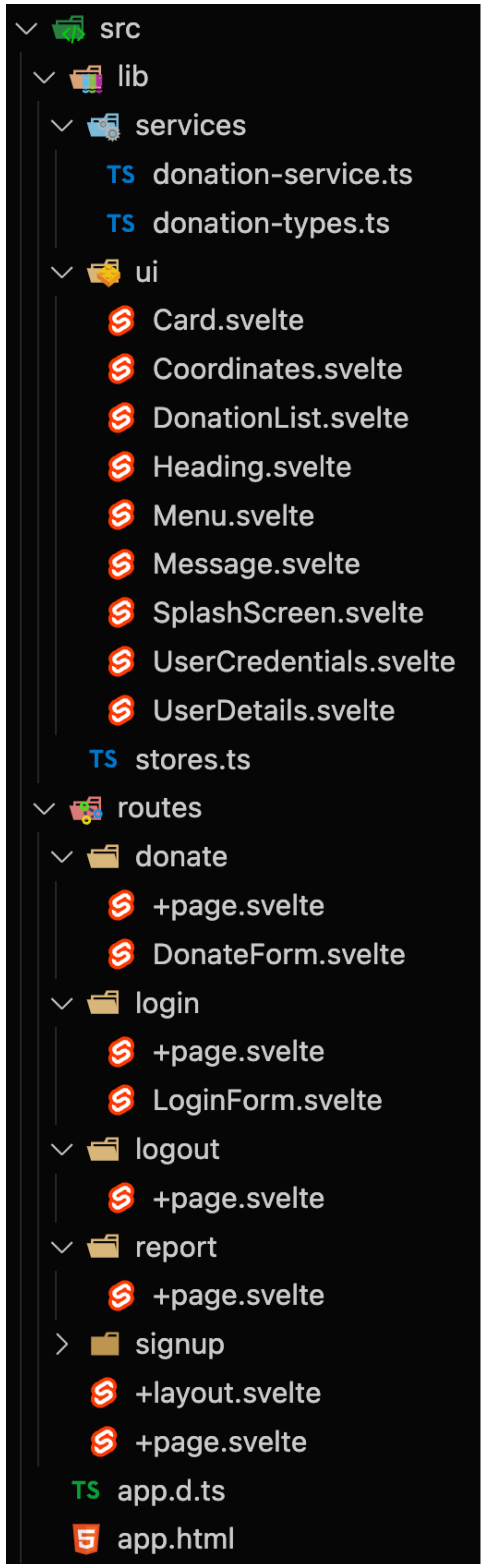
Incorporate authentication into Svelte application

Svelte Authentication

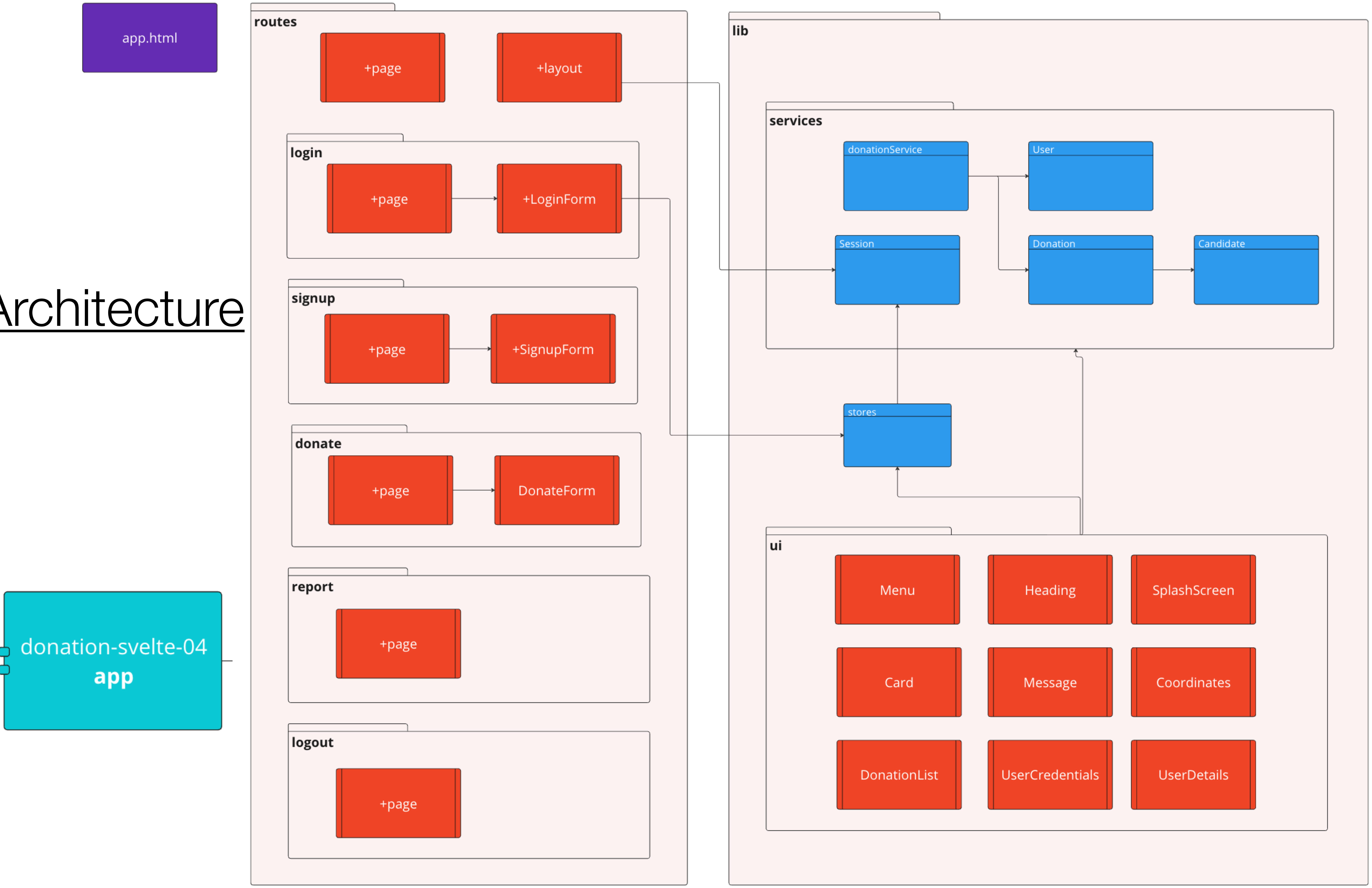


- Attach to the Donation-svelte API
- Signup / Login using Users API

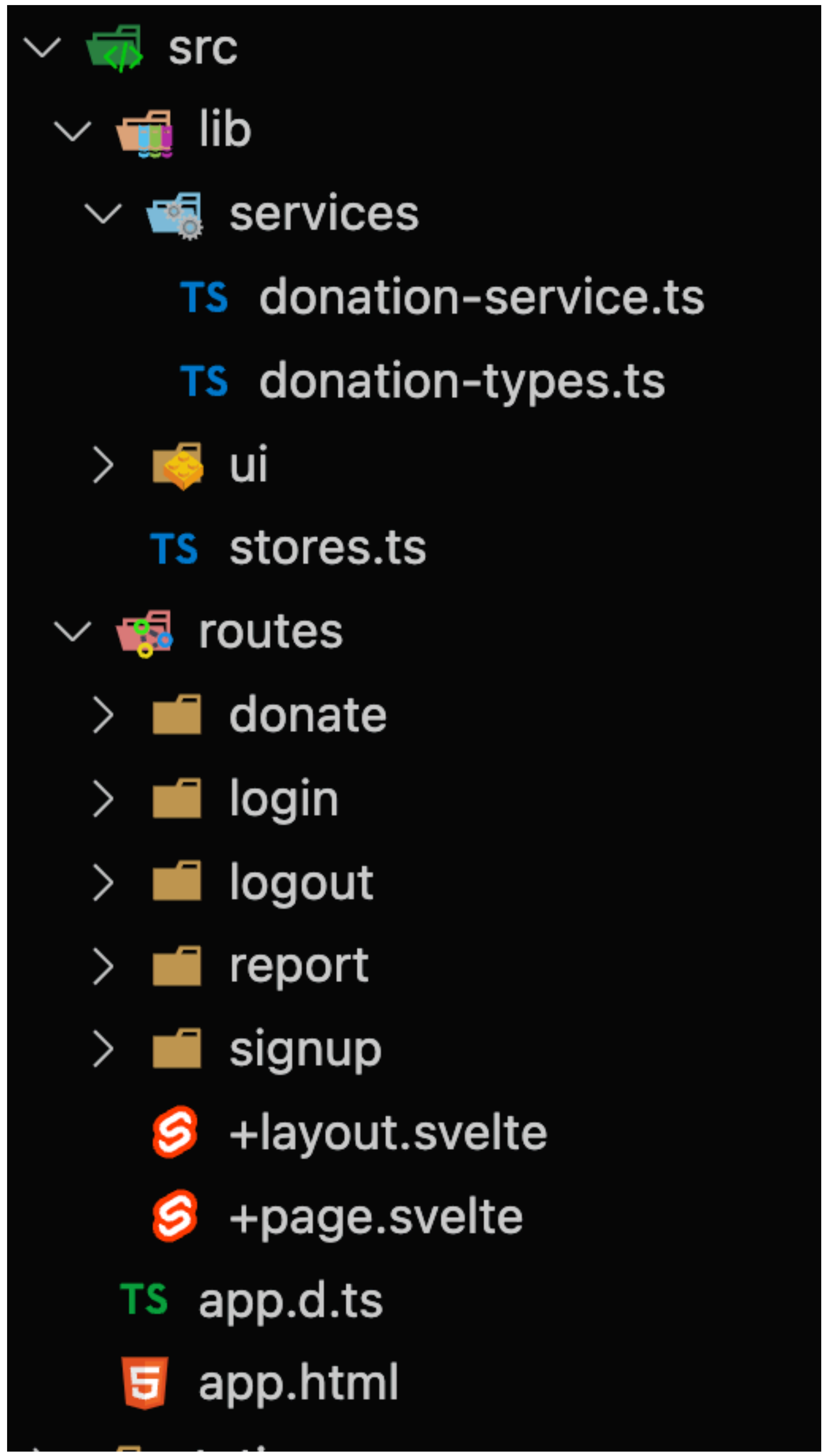
Architecture



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



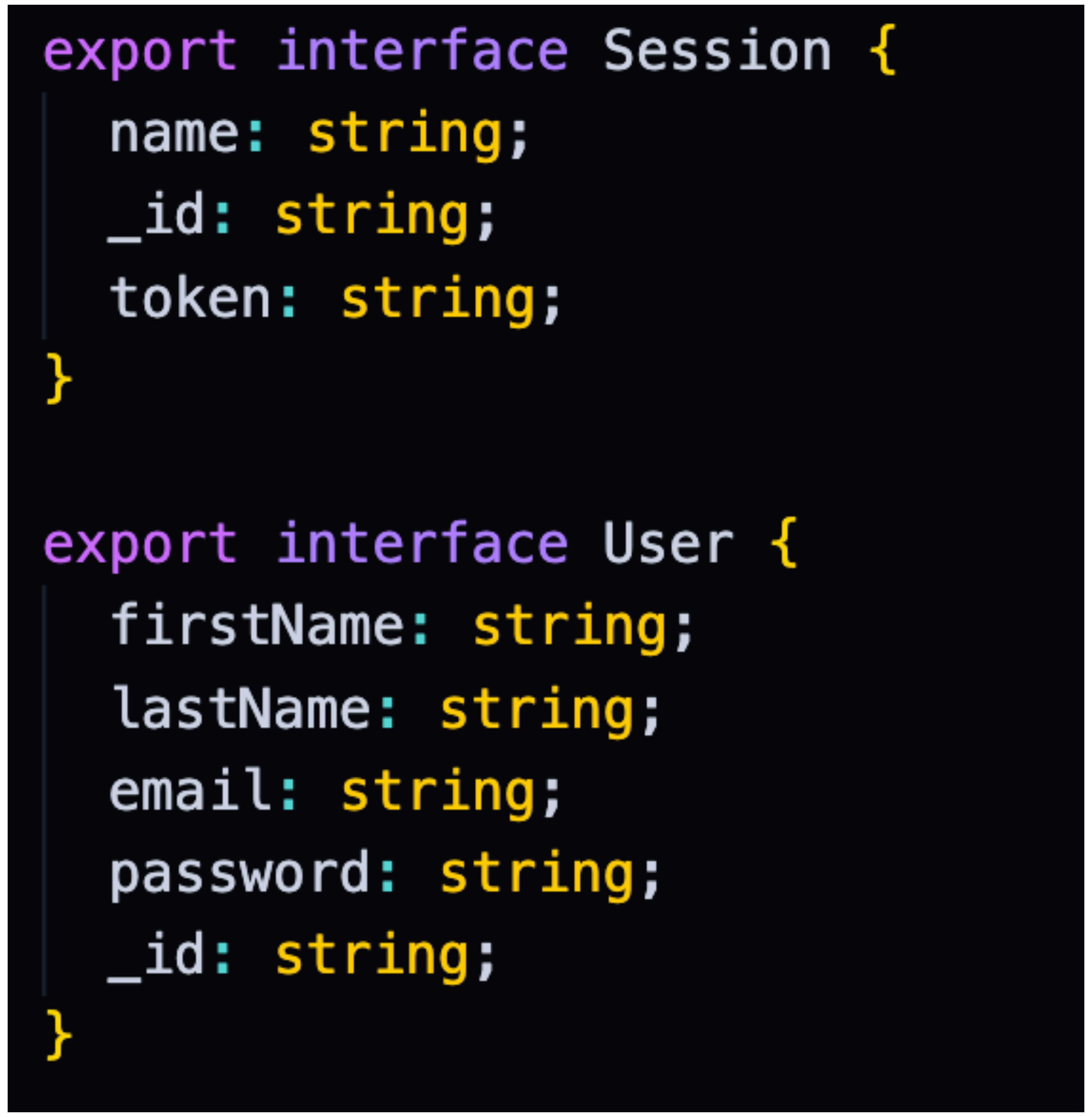
Donation-types



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

Session: logged in user name, id + JWT token

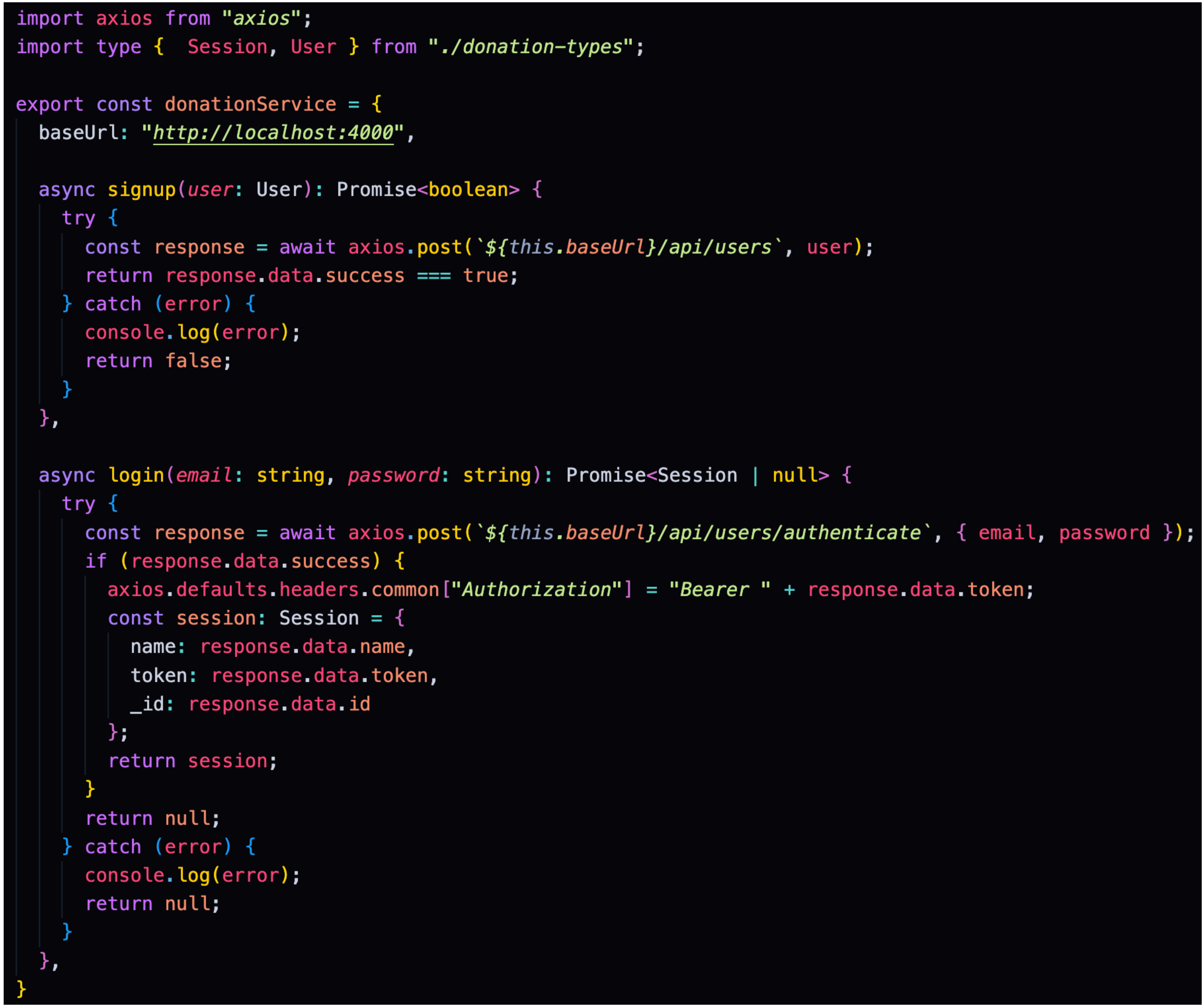
User: user details



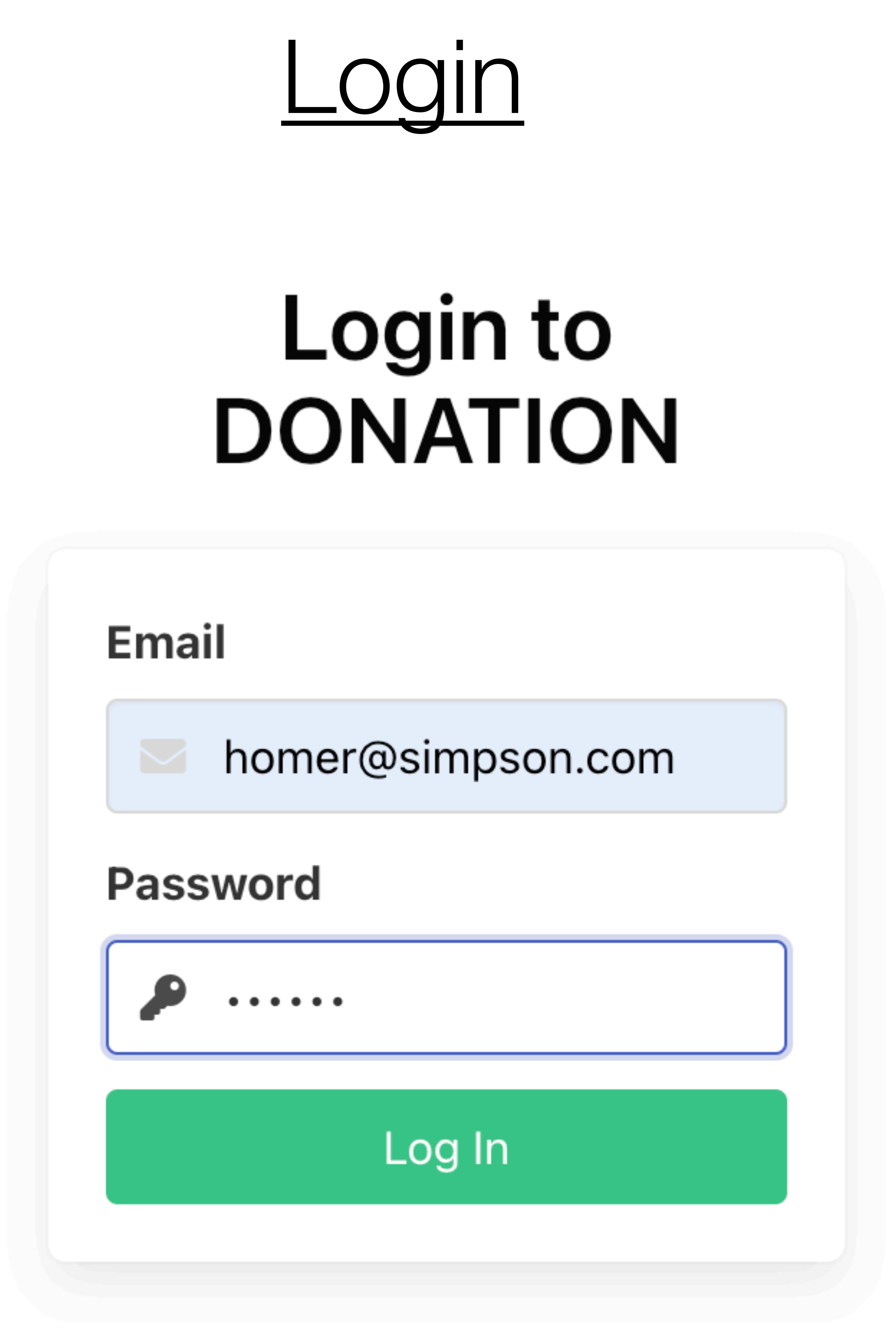
```
export interface Session {
  name: string;
  _id: string;
  token: string;
}

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

donationService

- Encapsulate
- baseUrl defines url of remote donation service
- login: invoke authenticate endpoint, return session object if successful
- signup: invoke signup up API.



```
import axios from "axios";
import type { Session, User } from "./donation-types";

export const donationService = {
  baseUrl: "http://localhost:4000",

  async signup(user: User): Promise<boolean> {
    try {
      const response = await axios.post(`${this.baseUrl}/api/users`, user);
      return response.data.success === true;
    } catch (error) {
      console.log(error);
      return false;
    }
  },

  async login(email: string, password: string): Promise<Session | null> {
    try {
      const response = await axios.post(`${this.baseUrl}/api/users/authenticate`, { email, password });
      if (response.data.success) {
        axios.defaults.headers.common["Authorization"] = "Bearer " + response.data.token;
        const session: Session = {
          name: response.data.name,
          token: response.data.token,
          _id: response.data.id
        };
        return session;
      }
      return null;
    } catch (error) {
      console.log(error);
      return null;
    }
  },
}
```


Login

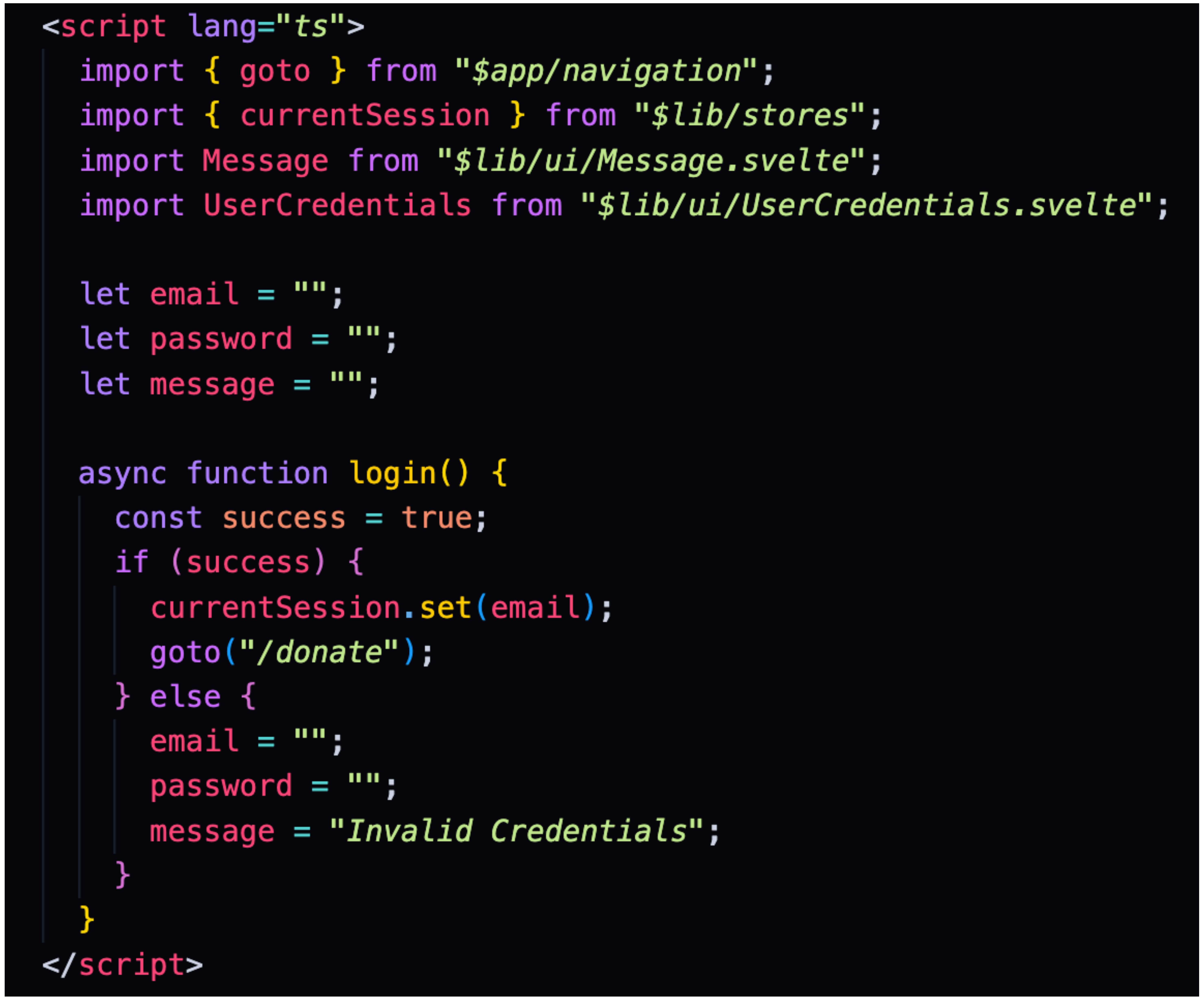
Login to DONATION

Email

homer@simpson.com

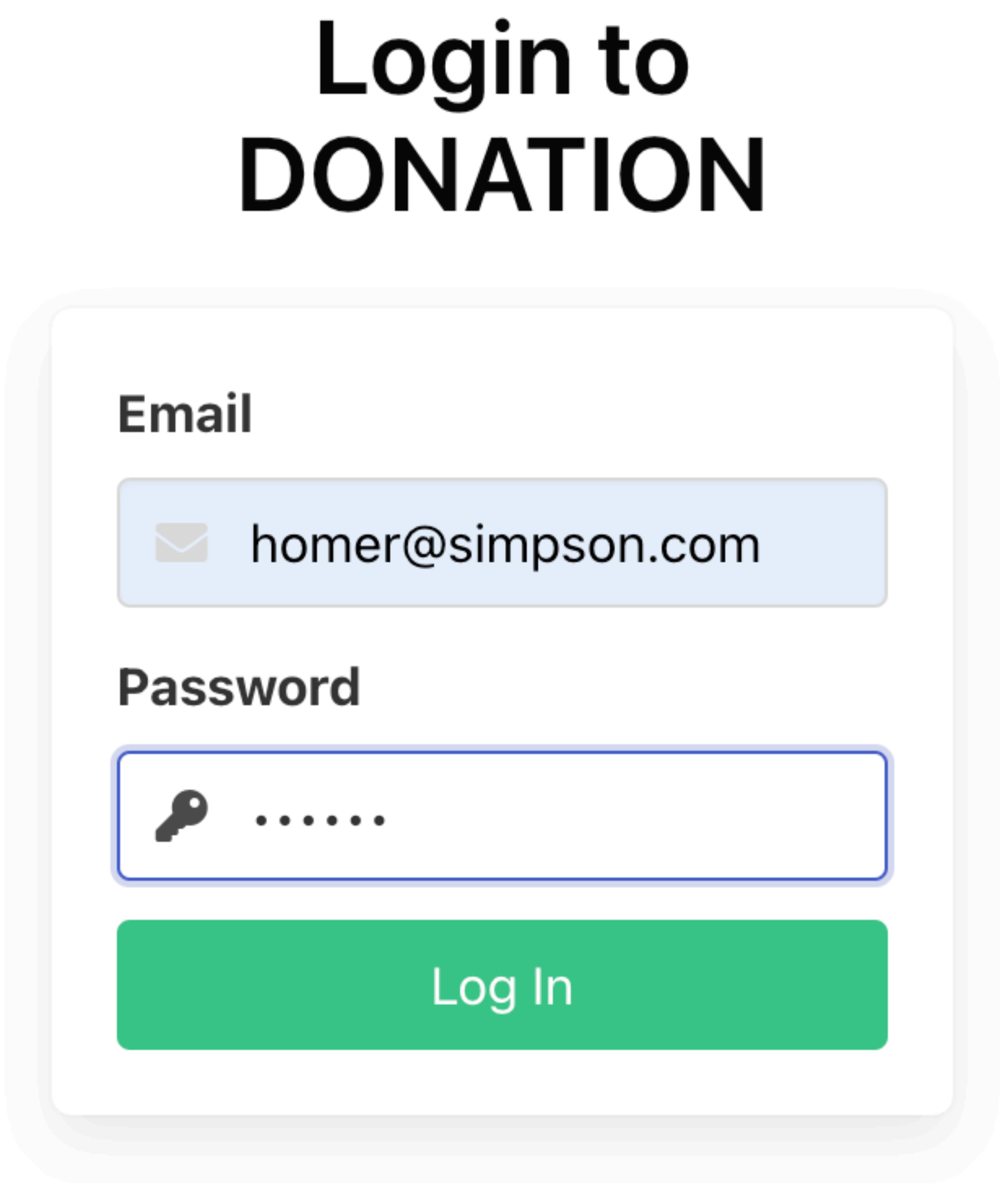
Password

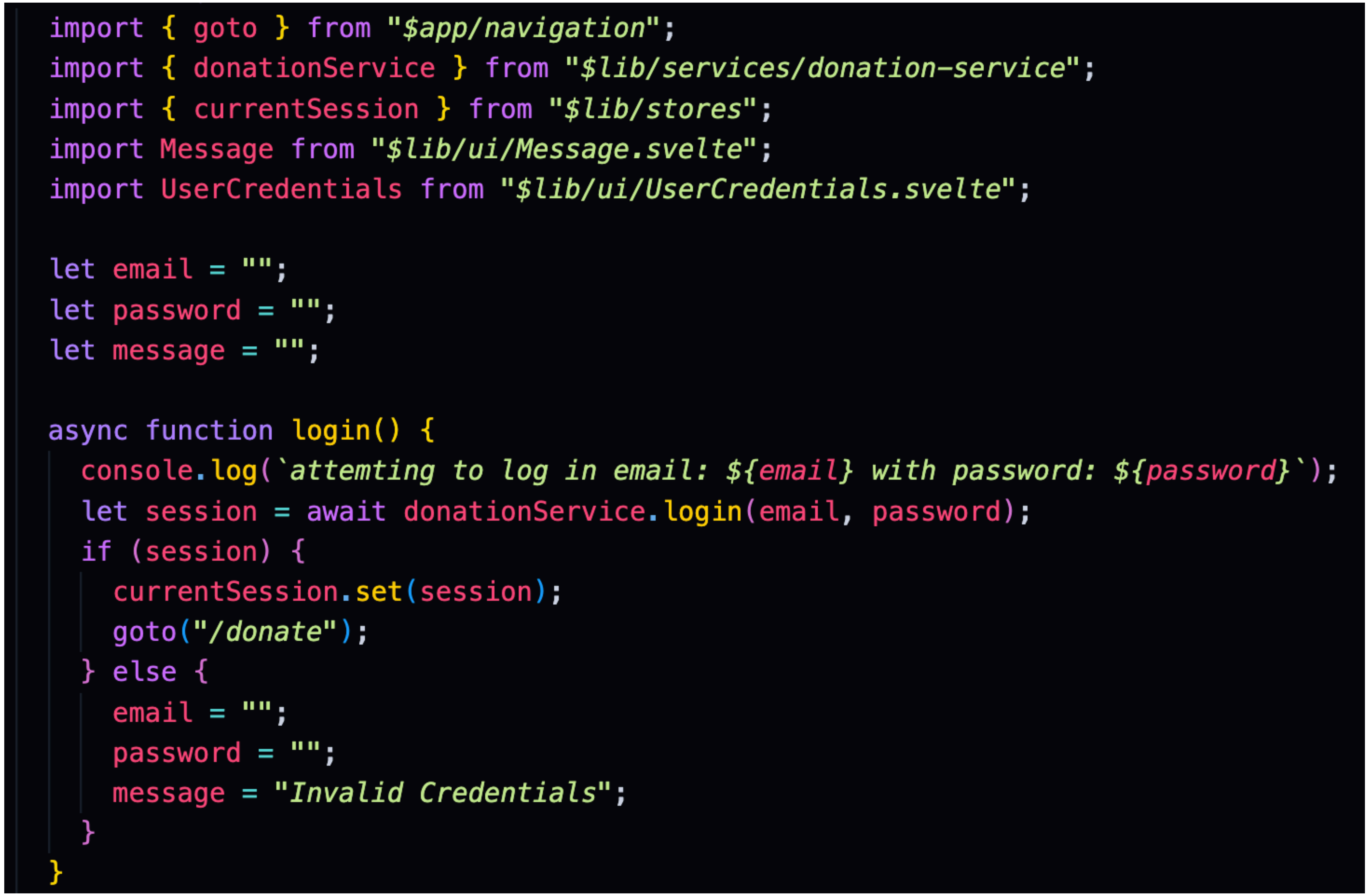
••••••


```
<script lang="ts">
  import { goto } from "$app/navigation";
  import { currentSession } from "$lib/stores";
  import Message from "$lib/ui/Message.svelte";
  import UserCredentials from "$lib/ui/UserCredentials.svelte";

  let email = "";
  let password = "";
  let message = "";

  async function login() {
    const success = true;
    if (success) {
      currentSession.set(email);
      goto("/donate");
    } else {
      email = "";
      password = "";
      message = "Invalid Credentials";
    }
  }
</script>
```

- Current version “goto” donate route when login button pressed

Login





```
import { goto } from "$app/navigation";
import { donationService } from "$lib/services/donation-service";
import { currentSession } from "$lib/stores";
import Message from "$lib/ui/Message.svelte";
import UserCredentials from "$lib/ui/UserCredentials.svelte";

let email = "";
let password = "";
let message = "";

async function login() {
  console.log(`attemting to log in email: ${email} with password: ${password}`);
  let session = await donationService.login(email, password);
  if (session) {
    currentSession.set(session);
    goto("/donate");
  } else {
    email = "";
    password = "";
    message = "Invalid Credentials";
  }
}
```

- Revised version to authenticate using Donation-hapi API

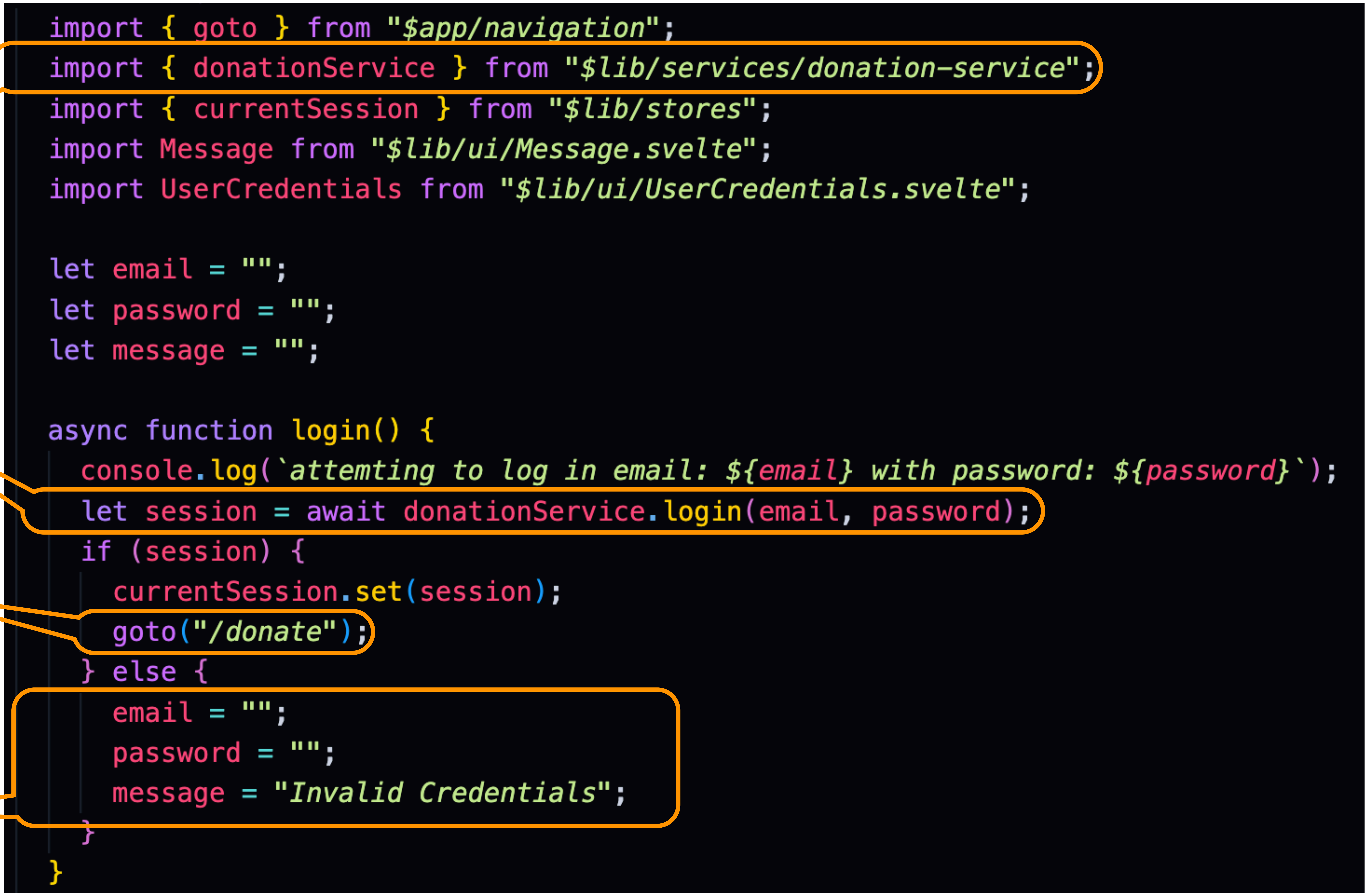
Login

Import a references to the donationService object.

When login pressed, attempt login

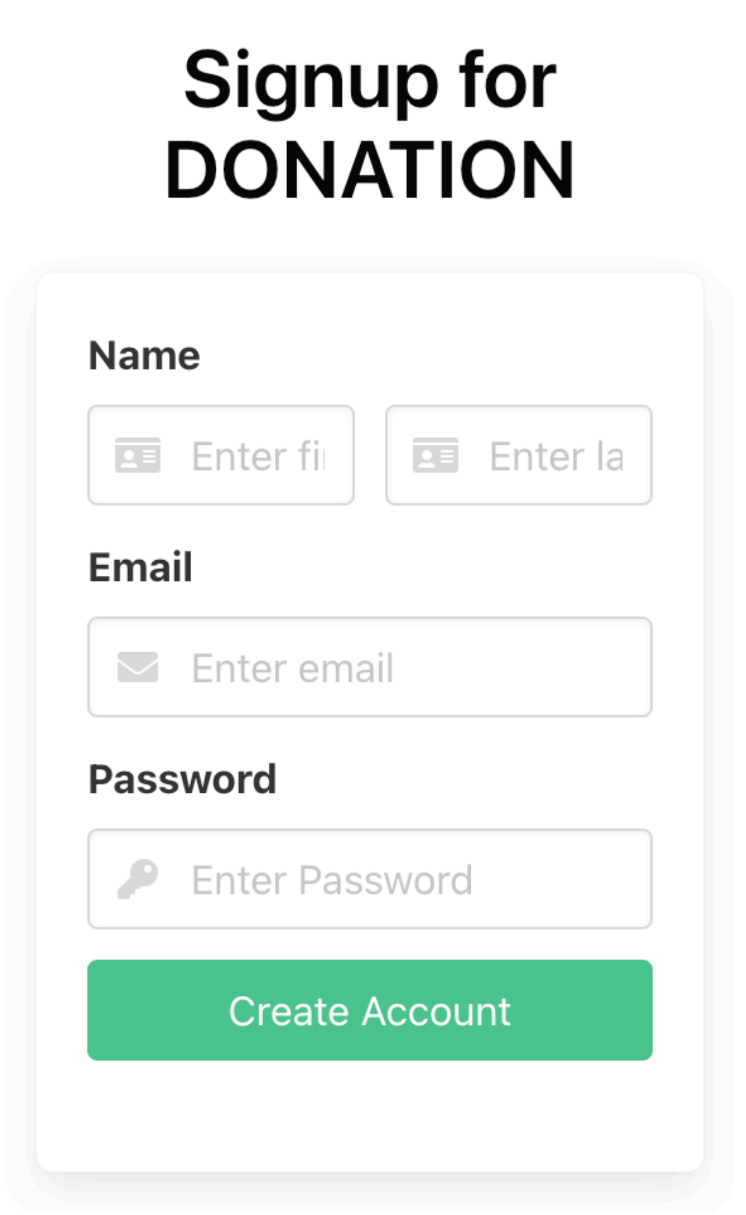
If successful doing, display **Donate** page

Otherwise, reset email/password to bank + display error message



```
import { goto } from "$app/navigation";
import { donationService } from "$lib/services/donation-service";
import { currentSession } from "$lib/stores";
import Message from "$lib/ui/Message.svelte";
import UserCredentials from "$lib/ui/UserCredentials.svelte";

let email = "";
let password = "";
let message = "";

async function login() {
  console.log(`attemting to log in email: ${email} with password: ${password}`);
  let session = await donationService.login(email, password);
  if (session) {
    currentSession.set(session);
    goto("/donate");
  } else {
    email = "";
    password = "";
    message = "Invalid Credentials";
  }
}
```

Signup


Signup for DONATION

Name

Enter fi

Enter la

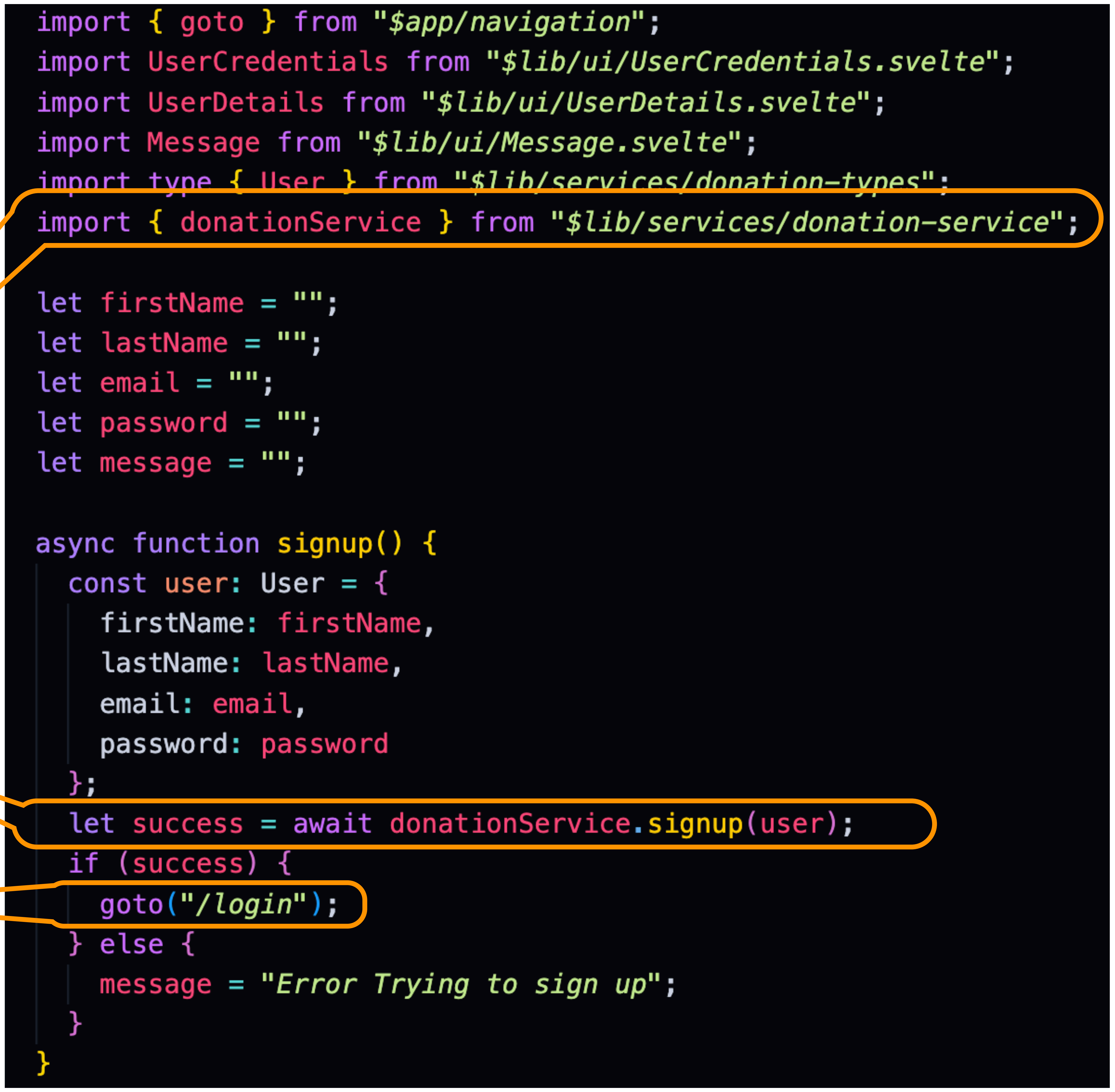
Email

Enter email

Password

Enter Password

Create Account

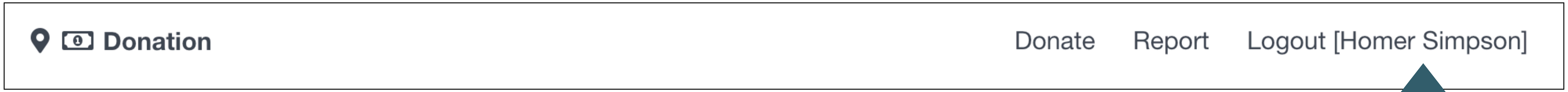

```
import { goto } from "$app/navigation";
import UserCredentials from "$lib/ui/UserCredentials.svelte";
import UserDetails from "$lib/ui/UserDetails.svelte";
import Message from "$lib/ui/Message.svelte";
import type { User } from "$lib/services/donation-types";
import { donationService } from "$lib/services/donation-service";

let firstName = "";
let lastName = "";
let email = "";
let password = "";
let message = "";

async function signup() {
  const user: User = {
    firstName: firstName,
    lastName: lastName,
    email: email,
    password: password
  };
  let success = await donationService.signup(user);
  if (success) {
    goto("/login");
  } else {
    message = "Error Trying to sign up";
  }
}
```

Import a reference to the donationService object.

Signup up new user

If successful doing, display **Main** page

Logged in User Indicator



Donation

Donate Report Logout [Homer Simpson]

- When user logged in…

... display user name in Menu bar

Stores

➡ **Stores:** A feature to allow unrelated components to talk to each other

Import *writeable,* create, initialise and export a **store** - from a .js file (not .svelte)

```
import { writable } from 'svelte/store'
export const username = writable('Guest')
```

In any component, import the store

```
<script>
import { username } from './store.js'
</script>
```

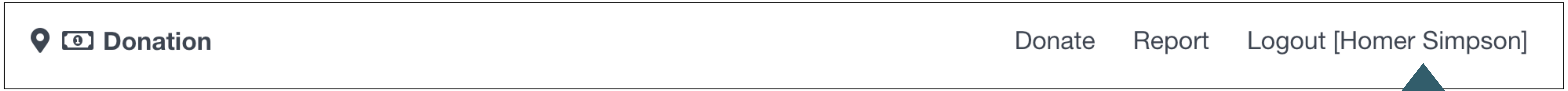
Update store (write)

```
username.set('new username')
```

Be notified if any other component has updated the store (read)

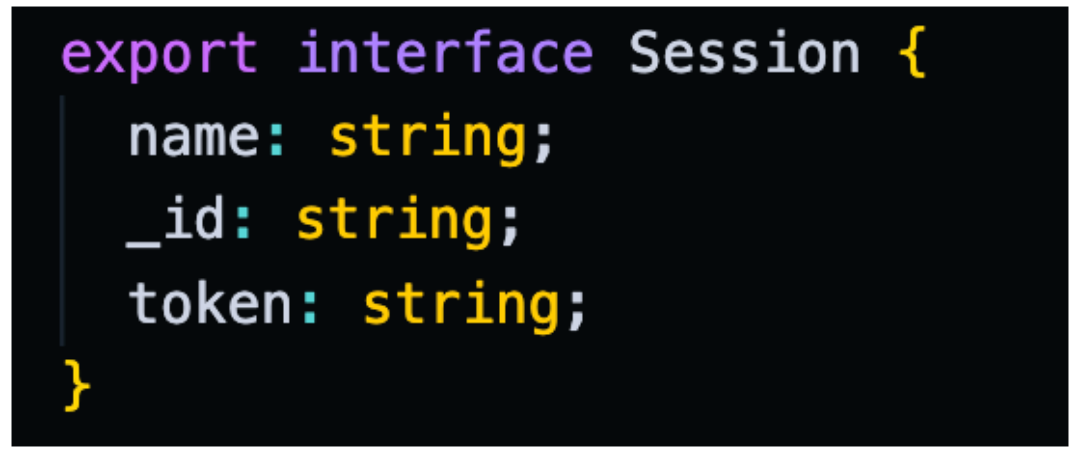
```
username.subscribe(newValue => {
  console.log(newValue)
})
```

Logged in User Indicator

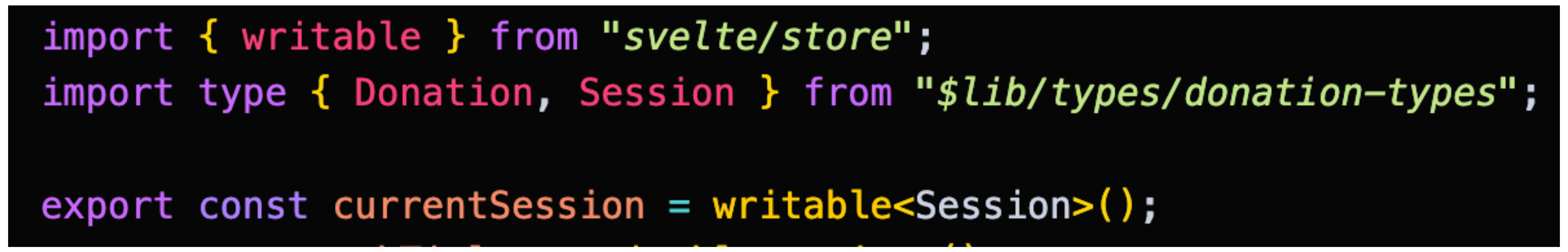


- When user logged in…

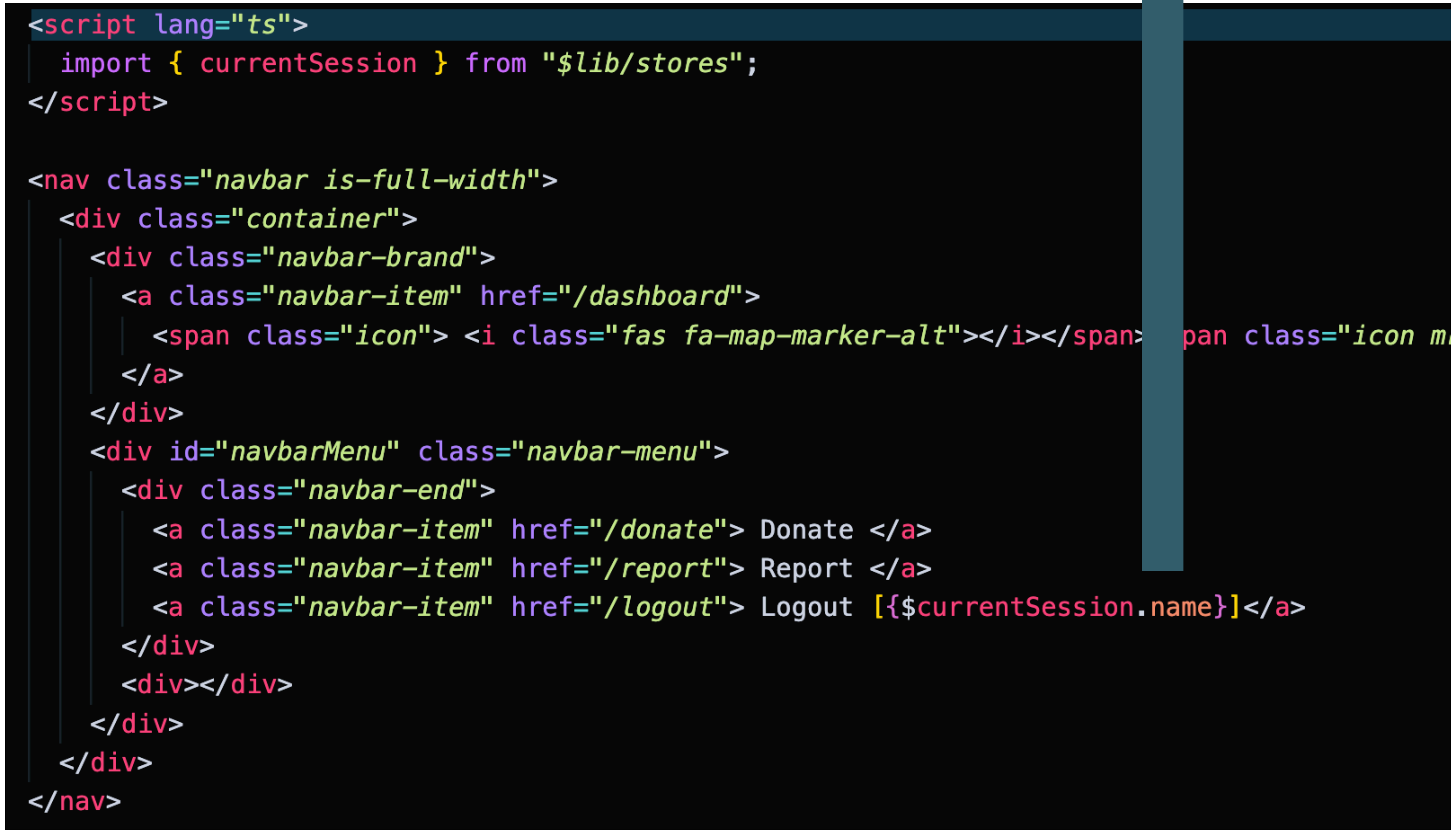
… display user name in Menu bar



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



```
import { writable } from "svelte/store";
import type { Donation, Session } from "$lib/types/donation-types";

export const currentSession = writable<Session>();
```

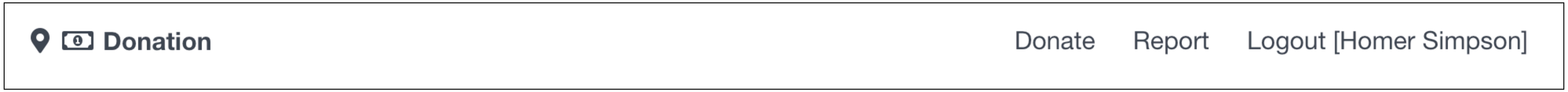


```
<script lang="ts">
  import { currentSession } from "$lib/stores";
</script>

<nav class="navbar is-full-width">
  <div class="container">
    <div class="navbar-brand">
      <a class="navbar-item" href="/dashboard">
        <span class="icon"> <i class="fas fa-map-marker-alt"></i></span> pan class="icon m
      </a>
    </div>
    <div id="navbarMenu" class="navbar-menu">
      <div class="navbar-end">
        <a class="navbar-item" href="/donate"> Donate </a>
        <a class="navbar-item" href="/report"> Report </a>
        <a class="navbar-item" href="/logout"> Logout [{$currentSession.name}]</a>
      </div>
      <div></div>
    </div>
  </div>
</nav>
```

- donationService create the session object

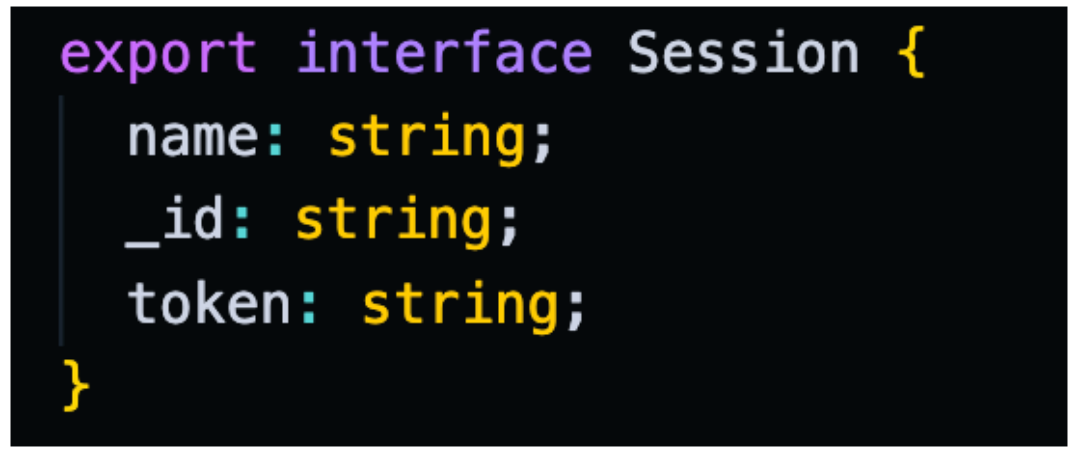
- LoginForm writes the session to the store

Logged in User Indicator

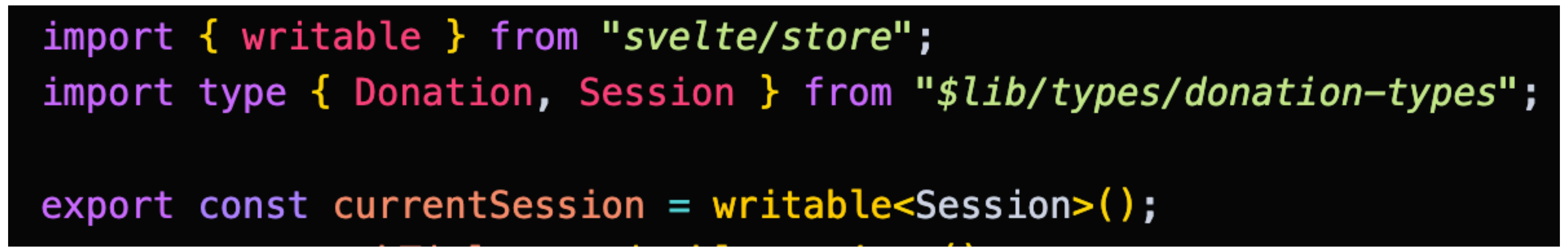


- When user logged in…

… display user name in Menu bar



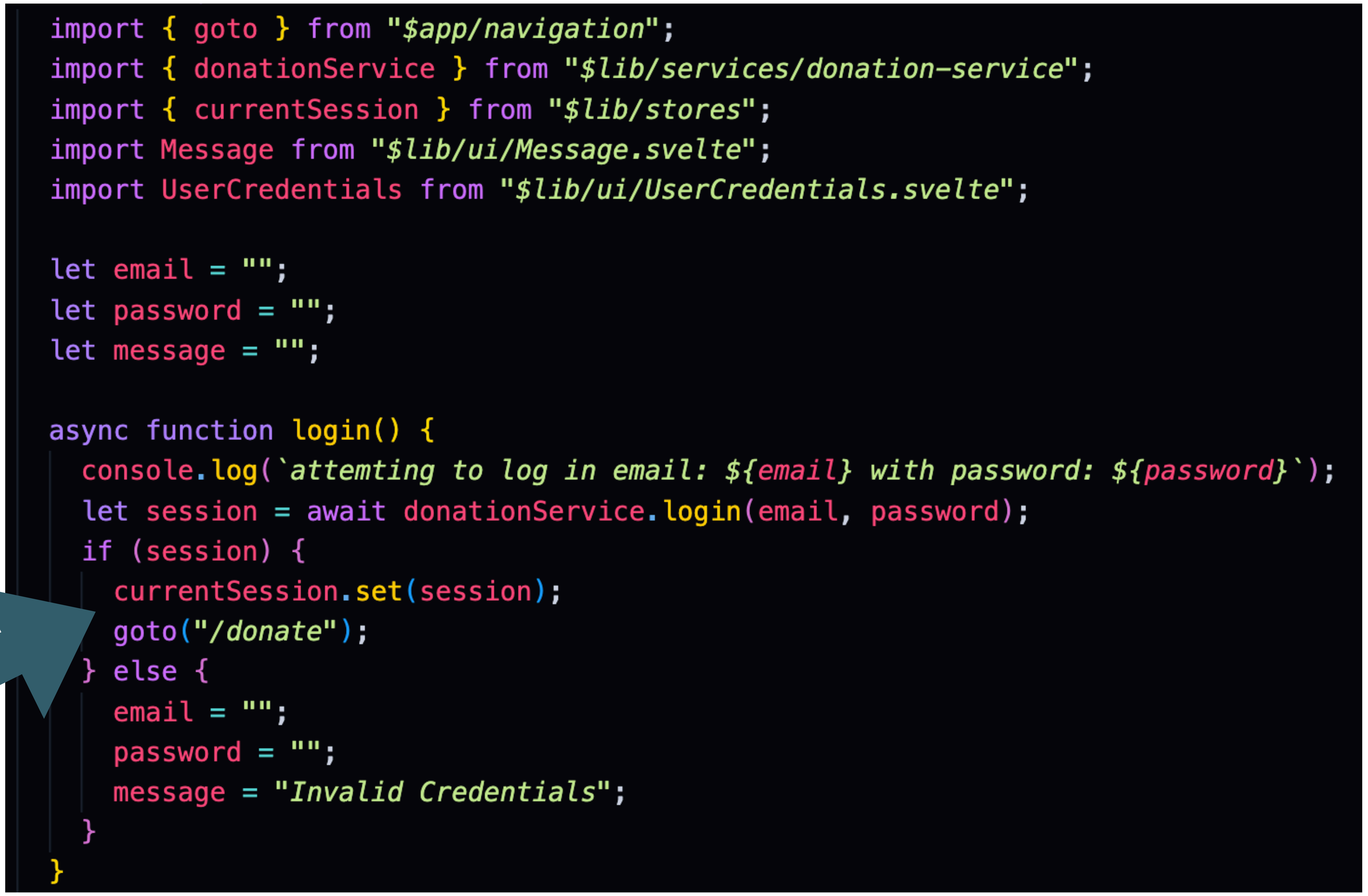
```
export interface Session {
  name: string;
  _id: string;
  token: string;
}
```



```
import { writable } from "svelte/store";
import type { Donation, Session } from "$lib/types/donation-types";

export const currentSession = writable<Session>();
```

- donationService create the session object

- LoginForm writes the session to the store



```
import { goto } from "$app/navigation";
import { donationService } from "$lib/services/donation-service";
import { currentSession } from "$lib/stores";
import Message from "$lib/ui/Message.svelte";
import UserCredentials from "$lib/ui/UserCredentials.svelte";

let email = "";
let password = "";
let message = "";

async function login() {
  console.log(`attemting to log in email: ${email} with password: ${password}`);
  let session = await donationService.login(email, password);
  if (session) {
    currentSession.set(session);
    goto("/donate");
  } else {
    email = "";
    password = "";
    message = "Invalid Credentials";
  }
}
```

Svelte Authentication

Incorporate authentication into Svelte application