UX 2 Plant App

Thomas Rhodes
459274825

Last Updated: May 20, 2022

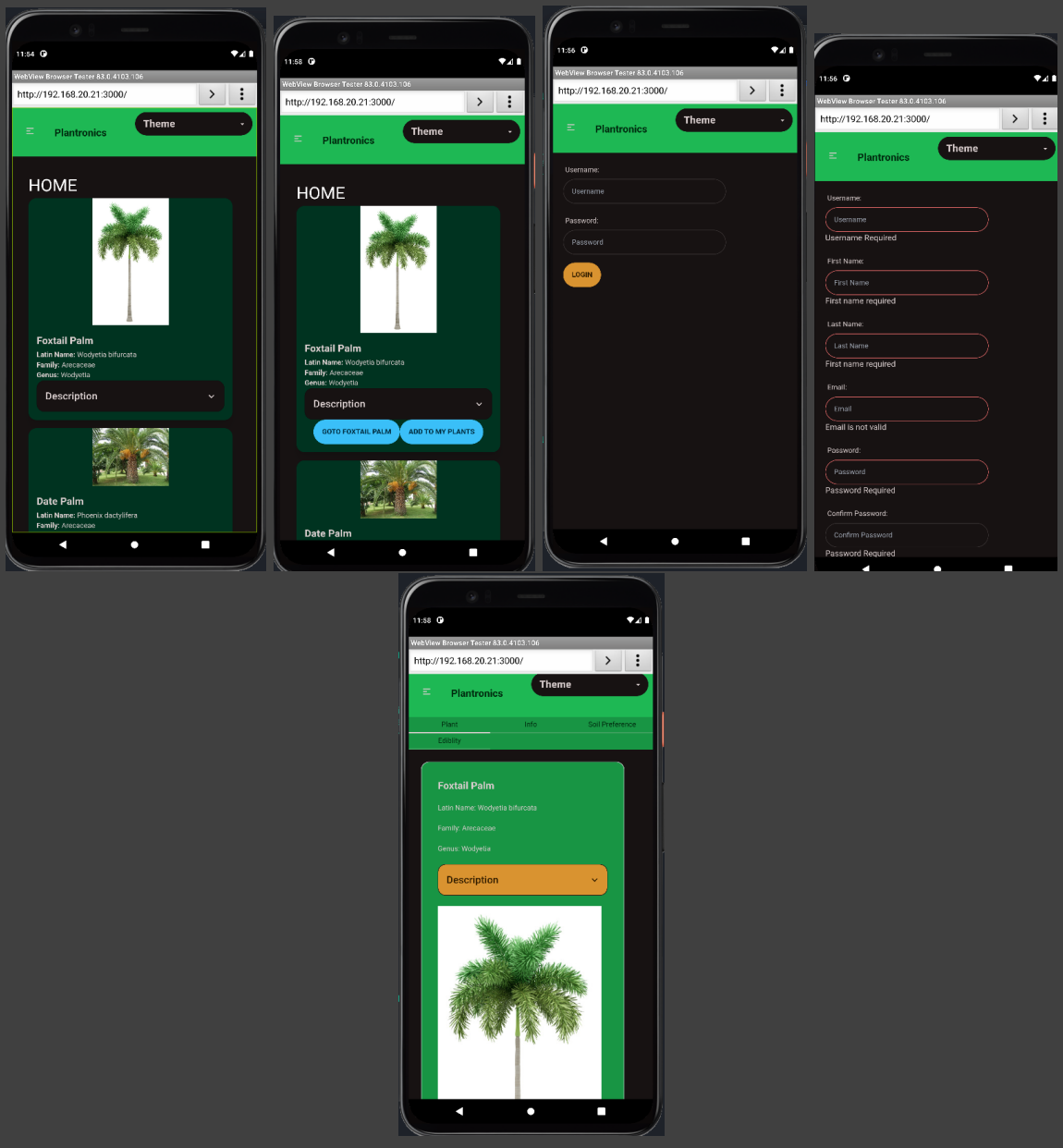
Contents

Part A

1 Implement project as a mobile only app

To showcase that this is infact a mobile first application, it was deemed necessary that a android emulator such as android-studio be used to take screenshots of the app in action.

Fig. 1: App homepage, homepage login page, registration page, plant slug page



1.1 What makes an app mobile first

A mobile first app in my opinion a mobile first application is the interactive parts, such as buttons, links, navbar and tabs are designed in a way that allows for people with large hands to happily and easily use the app, without disadvantaging users with smaller hands, And text should we center where possible for makes sense to allow for smoother scaling

2 Manifest file contains the relevant info required by the app

manifest.json:

```
{
  "name": "Plantronics",
  "short_name": "Plant",
  "theme_color": "#3367d6",
  "background_color": "#3367d6",
  "display": "standalone",
  "orientation": "portrait",
  "scope": "/",
  "start_url": "/",
  "icons": [
    {
      "src": "images/icons/icon-72x72.png",
      "sizes": "72x72",
      "type": "image/png",
      "purpose": "maskable"
    },
    {
      "src": "images/icons/icon-96x96.png",
      "sizes": "96x96",
      "type": "image/png"
    },
    {
      "src": "images/icons/icon-128x128.png",
      "sizes": "128x128",
      "type": "image/png"
    },
    {
      "src": "images/icons/icon-144x144.png",
      "sizes": "144x144",
      "type": "image/png"
    },
    {
      "src": "images/icons/icon-152x152.png",
      "sizes": "152x152",
      "type": "image/png"
    },
    {
      "src": "images/icons/icon-192x192.png",
      "sizes": "192x192",
      "type": "image/png"
    },
    {
      "src": "images/icons/icon-384x384.png",
      "sizes": "384x384",
```

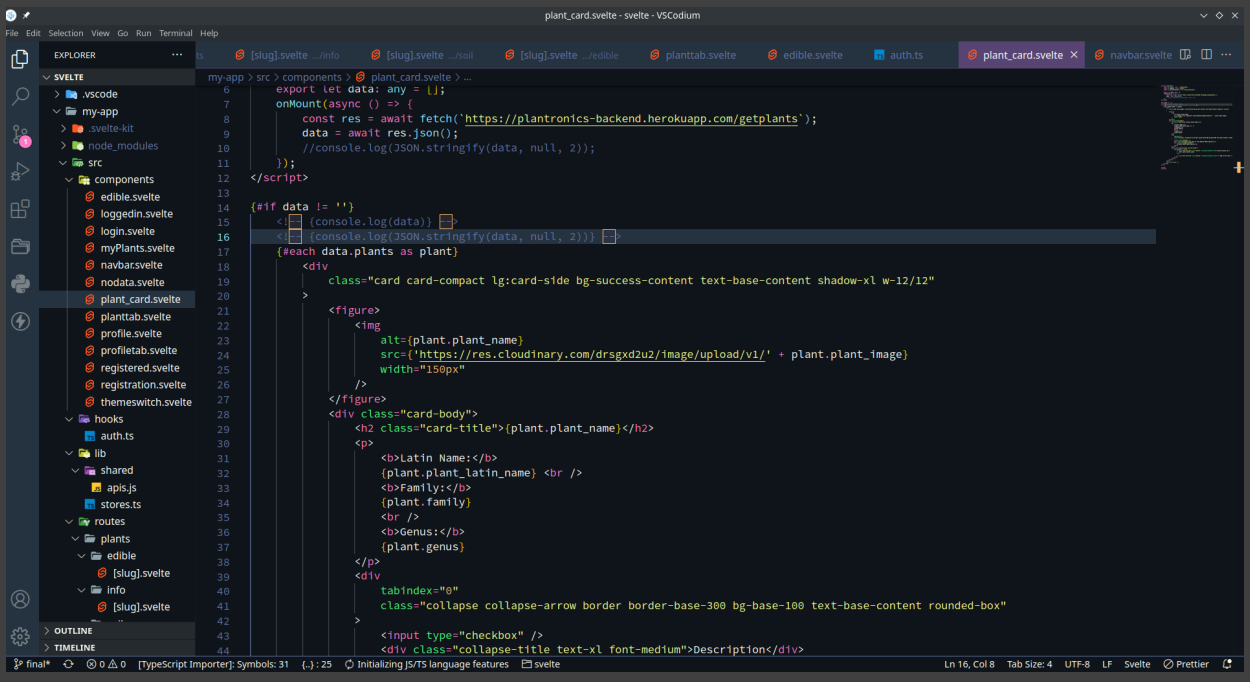
```
      "type": "image/png"
    },
    {
      "src": "images/icons/icon-512x512.png",
      "sizes": "512x512",
      "type": "image/png"
    }
  ]
}
```

We meet all the requirements for a pwa and the optional extras.

3 Use of custom design components that are initially void of user requested data

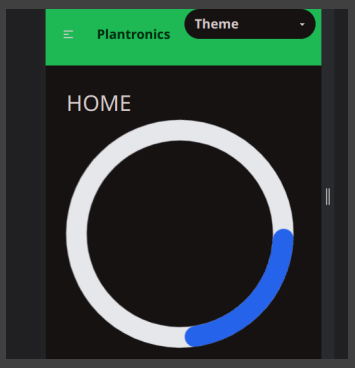
In Plantronics there are a a few components which are initially void of user requested data, They are as follows: edible.svelte, myPlants.svelte, plant_card.svelte and profile.svelte. which are all stored in the src/components.

Shown below is the plant_card.svelte component:



4 Third party interactive components used in implementation

The spinner is a interactive component from daisyui.



5 User selectable themes

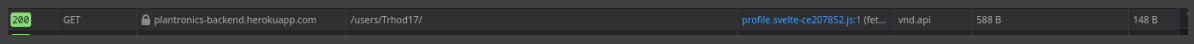
In the navbar there is a button that by default should be labeled Forest (which is the default theme, although it might be labeled theme which happens sometimes), which when click presents the user with a dropdown with a large amount of available themes (may some minor color issues on some themes), The default themes is full of rich greens/browns and earthy tones, but users are free to pick whichever theme they want. The themes themselves are from daisyui, and the theme selector is achieved by using a package called theme-select which handles the logic and is recommened by daisyui.

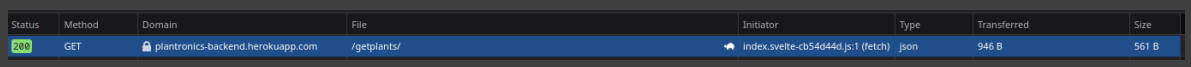
Part B

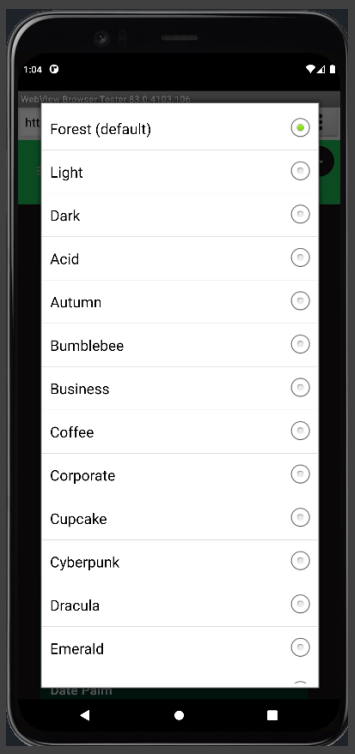
6 Form validation of all input fields

In plantronics at the time of writing this the components that contain input fields are login.svelte, registration.svelte components, they use both html5 and typescript/felte+yup validation

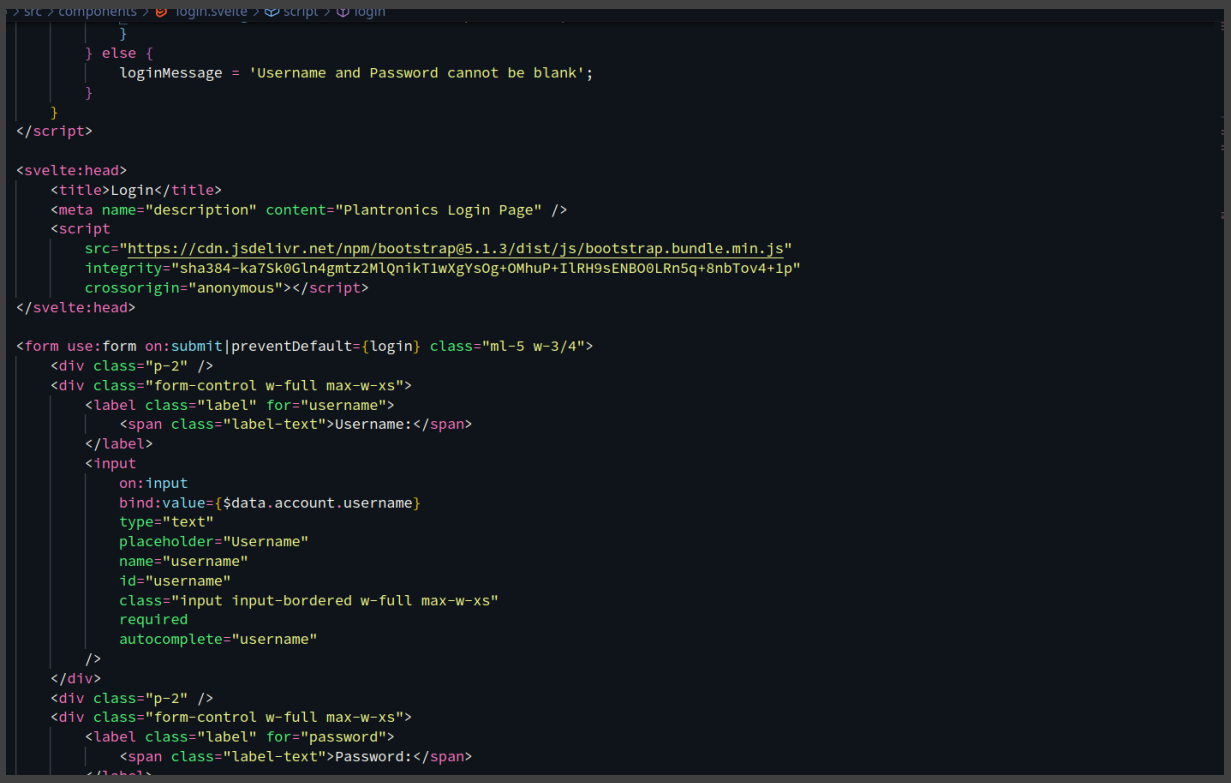
7 AJAX Implemented for at least 4 get and 4 post requests

- Get Request to get user info for profile in src/components/profile.svelte
- 

- Get Request to get all plants for index page in src/routes/index.svelte
- 



- Get Request to get userplants on the profile page in src/components/myPlants.svelte
- Get Request to get specific plant for plant page in src/routes/plants/[slug].svelte
- Post Request to obtain a authentication token in login page in src/hooks/auth.ts
- Post Request to obtain specific plant info for info page in in src/routes/plants/info/[slug].svelte
- Post Request to create a account and obtain authentication token in registration page in src/hooks/auth.ts
- Post Request to get specific edible info for edible page in src/routes/edible/[slug].svelte

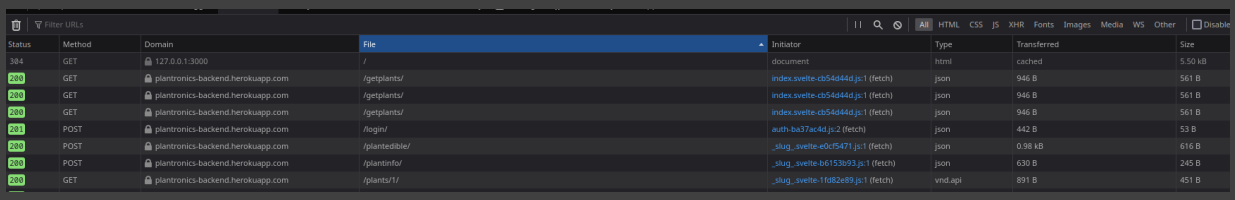


```
> src > components > login.svelte > script > login
            }
        } else {
            loginMessage = 'Username and Password cannot be blank';
        }
    }
</script>

<svelte:head>
    <title>Login</title>
    <meta name="description" content="Plantronics Login Page" />
    <script
        src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"
        integrity="sha384-ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p"
        crossorigin="anonymous"></script>
</svelte:head>

<form use:form on:submit|preventDefault={login} class="ml-5 w-3/4">
    <div class="p-2" />
    <div class="form-control w-full max-w-xs">
        <label class="label" for="username">
            <span class="label-text">Username:</span>
        </label>
        <input
            on:input
            bind:value={$data.account.username}
            type="text"
            placeholder="Username"
            name="username"
            id="username"
            class="input input-bordered w-full max-w-xs"
            required
            autocomplete="username"
        />
    </div>
    <div class="p-2" />
    <div class="form-control w-full max-w-xs">
        <label class="label" for="password">
            <span class="label-text">Password:</span>
```

8 Data interchange format between app and RESTful web service using json

The communications between the front and back ends utilise Json as means of sending and recieving json, which can be seen in the network tab and in the source code

Fig. 2: App homepage, homepage login page, registration page, plant slug page



Status	Method	Domain	File	Initiator	Type	Transferred	Size
304	GET	127.0.0.1:3000	/	document	html	cached	5.50 kB
200	GET	plantronics-backend.herokuapp.com	/getplants/	index.svelte-cb54d44d.js:1 (fetch)	json	946 B	561 B
200	GET	plantronics-backend.herokuapp.com	/getplants/	index.svelte-cb54d44d.js:1 (fetch)	json	946 B	561 B
200	GET	plantronics-backend.herokuapp.com	/getplants/	index.svelte-cb54d44d.js:1 (fetch)	json	946 B	561 B
201	POST	plantronics-backend.herokuapp.com	/login/	auth-ba37ac4d.js:2 (fetch)	json	442 B	53 B
200	POST	plantronics-backend.herokuapp.com	/plantedible/	_slug_.svelte-e0cf5471.js:1 (fetch)	json	0.98 kB	616 B
200	POST	plantronics-backend.herokuapp.com	/plantinfo/	_slug_.svelte-b6153b93.js:1 (fetch)	json	630 B	245 B
200	GET	plantronics-backend.herokuapp.com	/plants/1/	_slug_.svelte-1fd82e89.js:1 (fetch)	vnd.api	891 B	451 B

In the front end when we are sending data to the server we first have to convert the data into the Json format using JSON.stringify() is used to convert the data into a format which the backend can accept and parse. The backend is built in Django, using the Django Rest Framework package to create the rest functionality, Django uses serializers to take convert the data to and from json in order to handle the various functions its used in.

9 Use of fetch api to implement AJAX communication

This application uses the default Fetch implementation the comes as a part of javascript, to send and recieve requests and data.

Fig. 3: Here is a screenshot of the getUserDetail method is src/hooks/auth.ts

```
export const getUserDetails = async (user: string, pass: string) => {
    //let data = JSON.stringify({username: user, password: pass});
    //console.log(user + ' ' + pass);

    let headersList = {
        "Accept": "*/*",
        "Content-Type": "application/api.vnd+json"
    }

    let bodyContent = JSON.stringify({
        "username": user,
        "password": pass
    });

    return fetch("https://plantronics-backend.herokuapp.com/login/", {
        method: "POST",
        body: bodyContent,
        headers: headersList
    }).then(function (response) {
        if (response.status != 201) {
            return 'error'
        } else {
            return response.text();
        }
    }).then(function (data) {
        return data;
    })
}
```

We can see the code to send the user credidentials to the backend to validate whether they are correct.

10 Localstorage demonstrated remembering user actions, and app reload is contextually remembered at least three (3)

Because i used svelte stores to store the username and token to localstorage, there is only 1 entry for the token and username rather then two seperate ones.

Fig. 4: Vivaldi (chrome based)

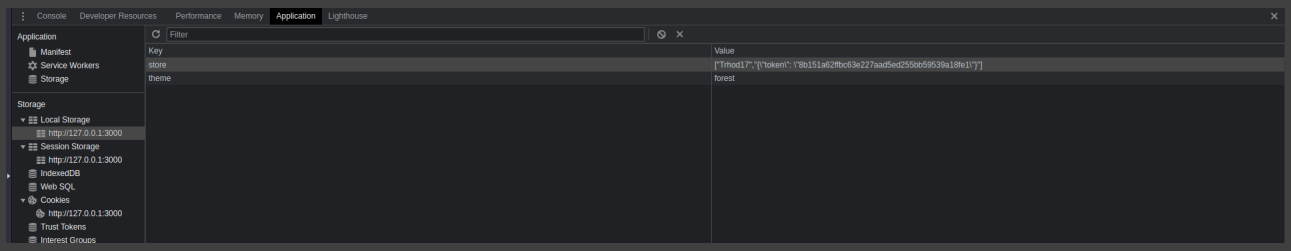
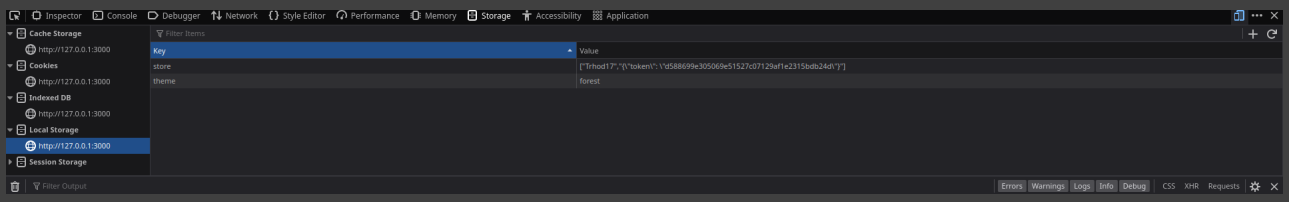
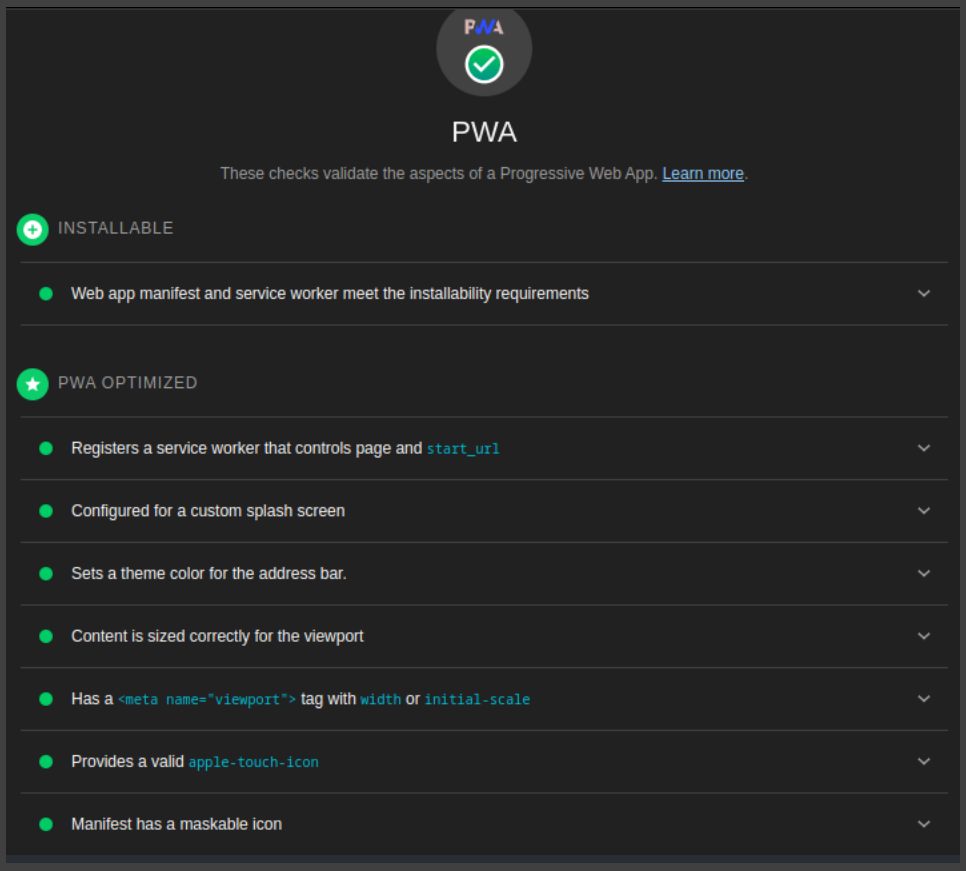


Fig. 5: Firefox



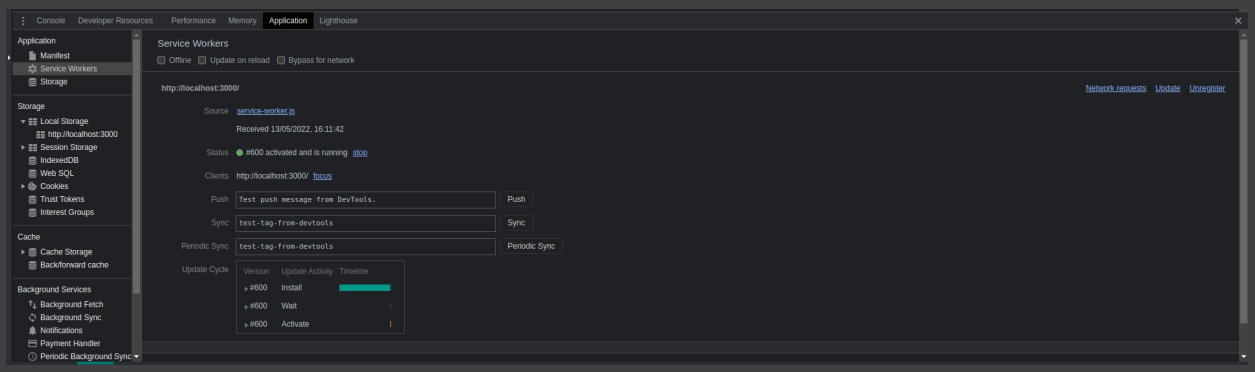
11 PWA audit complete under audit tab

Successful pwa audit in Vivaldi (chrome based)

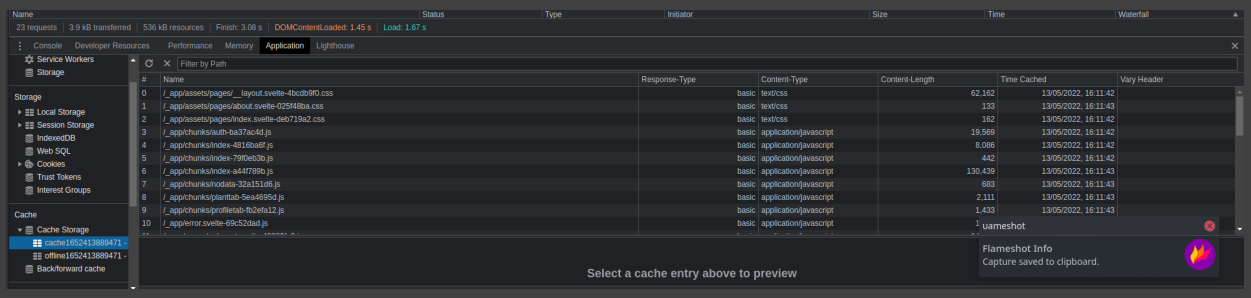


12 Service Worker to cache HTML/CSS/JS objects in-browser, and able to load without the network being present

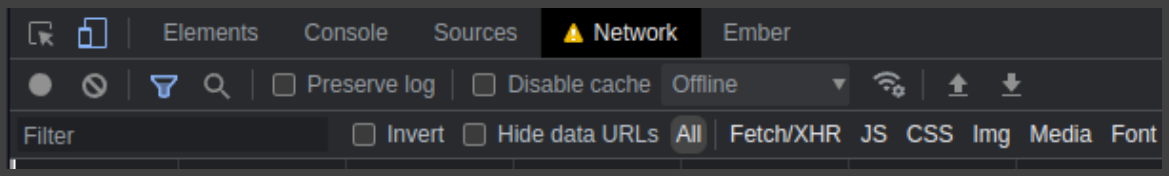
The service is currently caching css/html/js and also image files i believe, Below shows confirmation that the service worker is working.



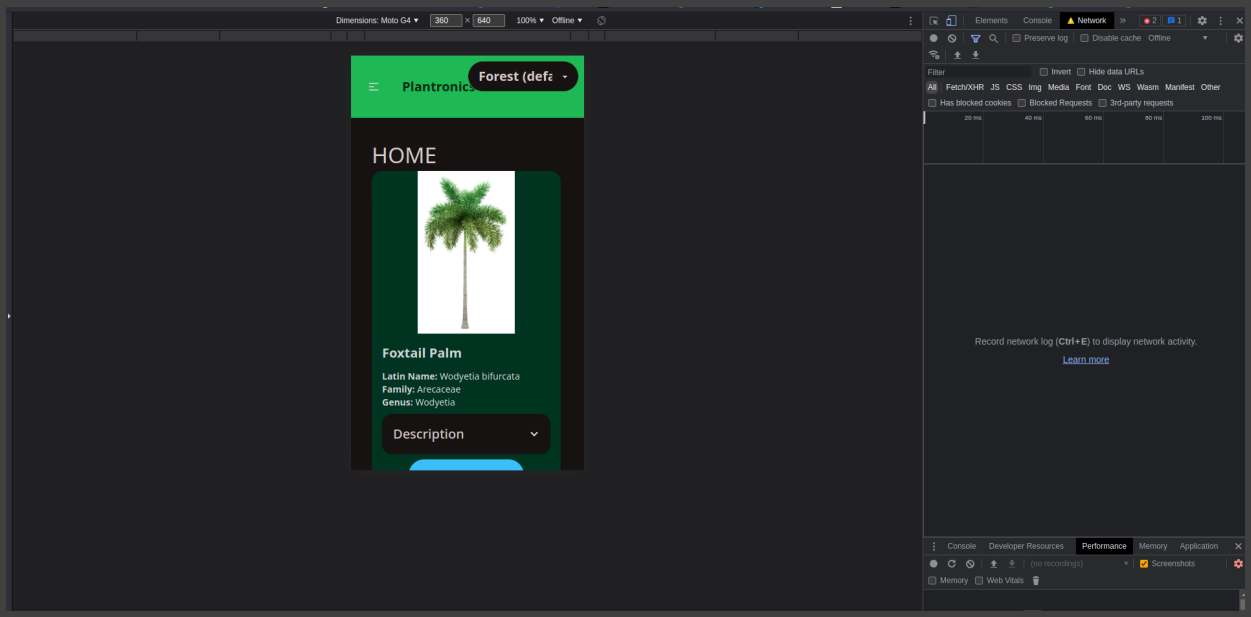
And here is the cache

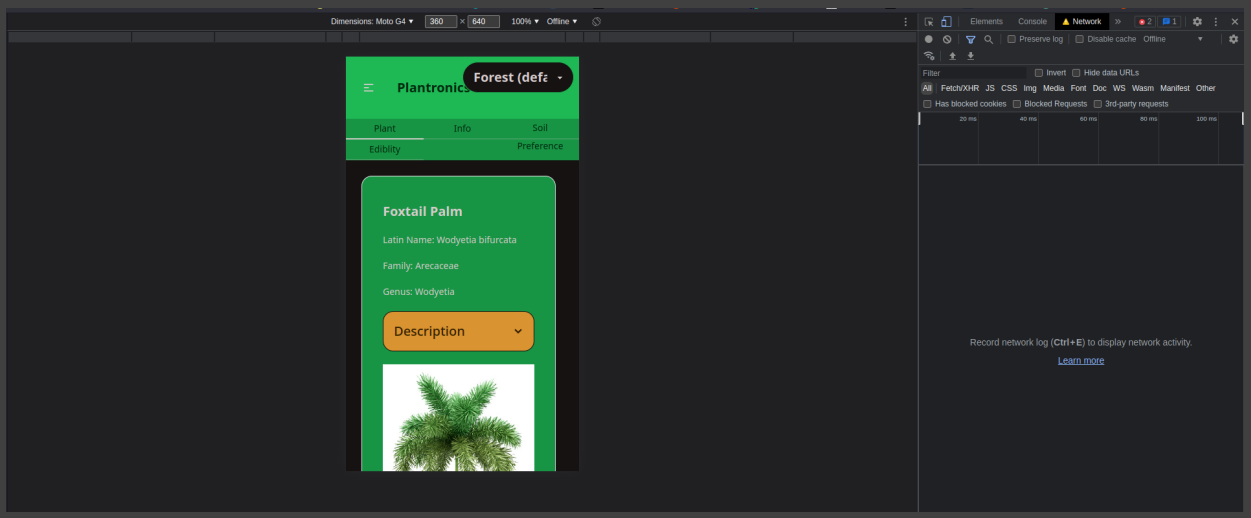


Using the throttling feature in the network tab we can make the website run offline



If we go to the home page and it loads with no issues including the images.
and the plant page for the foxtail palm also loads with no issues.





Part C

13 Icon for app in manifest

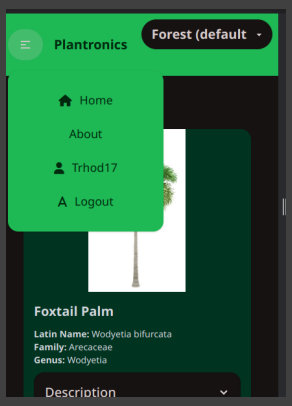
In the manifest json there are 8 icons ranging from 72x72 to 512x512, including 192x192 for apple devices.

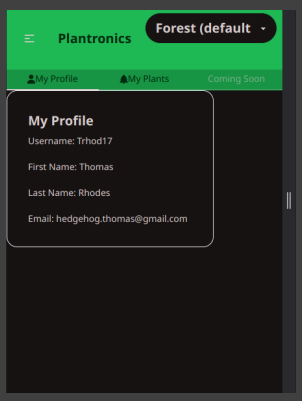
```
"icons": [
  {
    "src": "images/icons/icon-72x72.png",
```

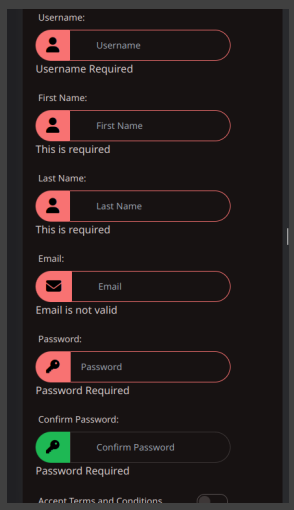
```
    "sizes": "72x72",
    "type": "image/png",
    "purpose": "maskable"
  },
  {
    "src": "images/icons/icon-96x96.png",
    "sizes": "96x96",
    "type": "image/png"
  },
  {
    "src": "images/icons/icon-128x128.png",
    "sizes": "128x128",
    "type": "image/png"
  },
  {
    "src": "images/icons/icon-144x144.png",
    "sizes": "144x144",
    "type": "image/png"
  },
  {
    "src": "images/icons/icon-152x152.png",
    "sizes": "152x152",
    "type": "image/png"
  },
  {
    "src": "images/icons/icon-192x192.png",
    "sizes": "192x192",
    "type": "image/png"
  },
  {
    "src": "images/icons/icon-384x384.png",
    "sizes": "384x384",
    "type": "image/png"
  },
  {
    "src": "images/icons/icon-512x512.png",
    "sizes": "512x512",
    "type": "image/png"
  }
]
```

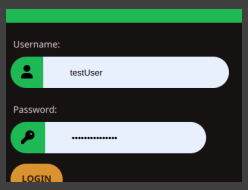
14 Generous use of glyphs found in layout framework for forms and menus

The navbar and profile page tabs, login, and registration forms have glyphs
font awesome was used for the glyphs


Plantronics
Forest (default
Home
About
Trhod17
Logout
Foxtail Palm
Latin Name: Wodyetia bifurcata
Family: Arecaceae
Genus: Wodyetia
Description

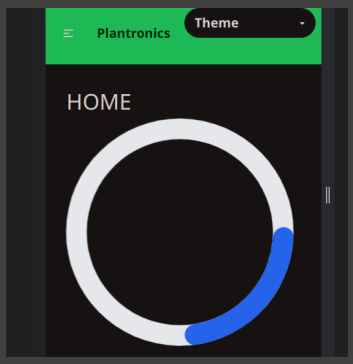

Plantronics
Forest (default
My Profile
My Plants
Coming Soon
My Profile
Username: Trhod17
First Name: Thomas
Last Name: Rhodes
Email: hedgehog.thomas@gmail.com


Username:
Username
Username Required
First Name:
First Name
This is required
Last Name:
Last Name
This is required
Email:
Email
Email is not valid
Password:
Password
Password Required
Confirm Password:
Confirm Password
Password Required



15 Temporary loading screen spinner overlay on display before JSON objects are rendered from Web Service

Currently only the index page has a loading spinner, because i didn't have time to implement it the rest of them

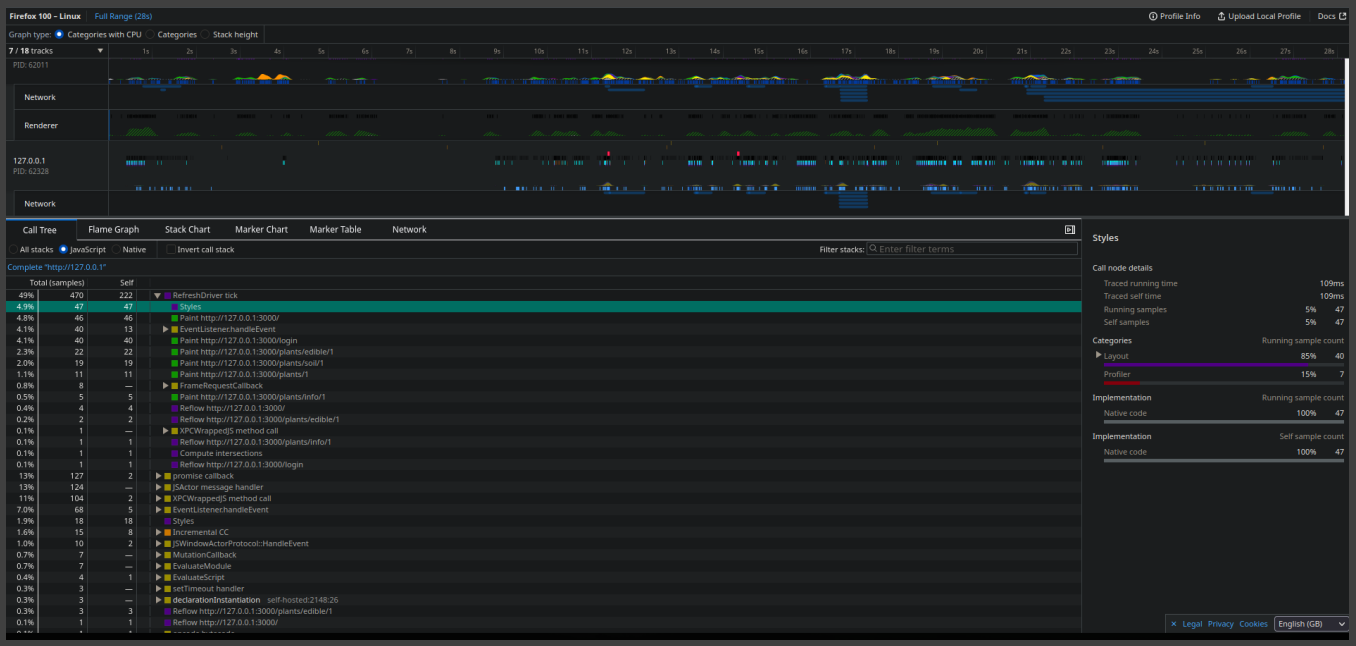


Part D

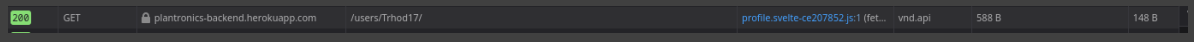
16 Screen shot of performance tab in devtools

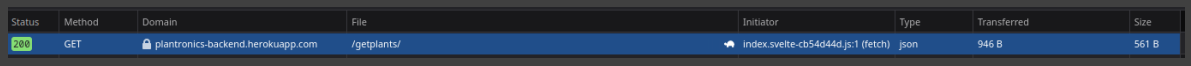
My internet was playing up a bit and the backend is currently hosted on heroku in america so some response times may be a tad slow.

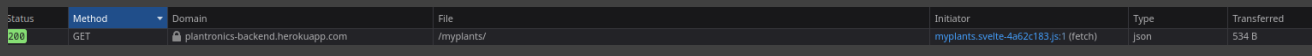
Fig. 6: The site sending and recieving data to/from the server from various pages

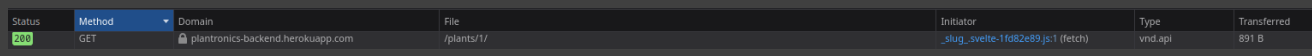


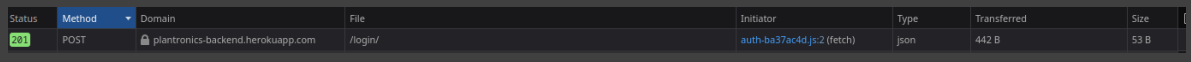
17 Screenshot showing ajax

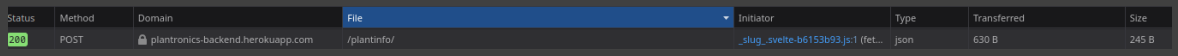
- Get Request to get user info for profile in src/components/profile.svelte
- 

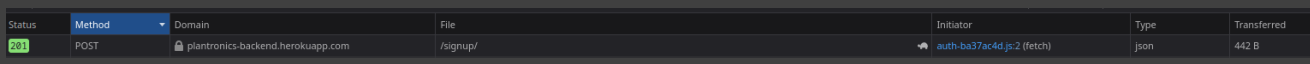
- Get Request to get all plants for index page in src/routes/index.svelte
- 

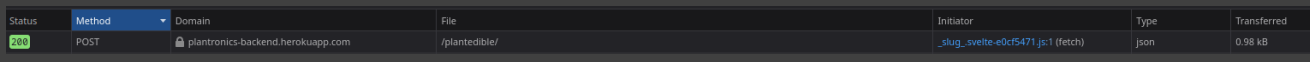
- Get Request to get userplants on the profile page in src/components/myPlants.svelte
- 

- Get Request to get specific plant for plant page in src/routes/plants/[slug].svelte
- 

- Post Request to obtain a authentication token in login page in src/hooks/auth.ts
- 

- Post Request to obtain specific plant info for info page in in src/routes/plants/info/[slug].svelte
- 

- Post Request to create a account and obtain authentication token in registration page in src/hooks/auth.ts
- 

- Post Request to get specific edible info for edible page in src/routes/edible/[slug].svelte
- 

18 Localstorage inside devtools to prove app works

Localstorage in both chrome and firefox

Fig. 7: Vivaldi (chrome based)

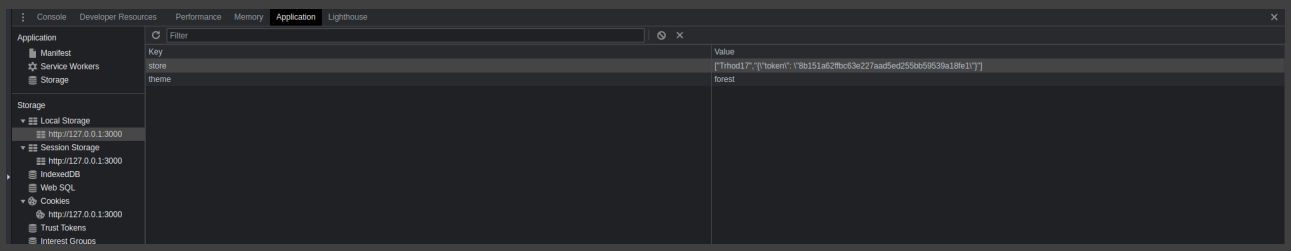
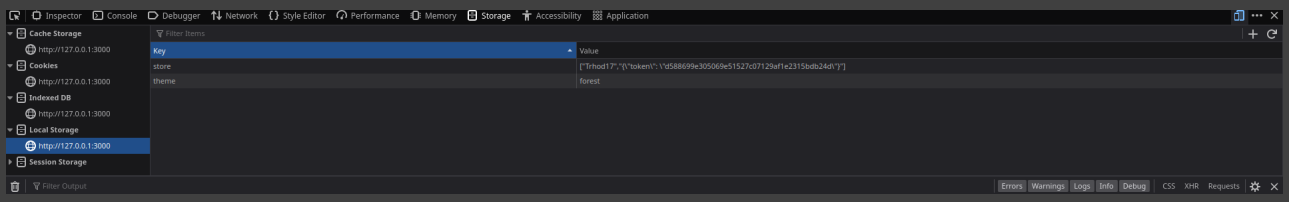


Fig. 8: Firefox



19 PWA audit complete under audit tab

Successful pwa audit in Vivaldi (chrome based)