DRAFT

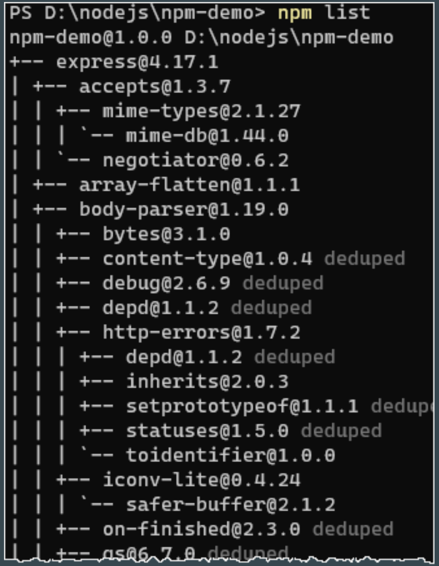
INDY-2
Package Update Notifier Project

•••

By: Jason Paek, Erik Smith, Nicholas Ott, Haris Yosufi

The Dependency Problem

Project Managers and Developers can have dozens of projects that can each depend on dozens or hundreds of packages, each with their own dependencies to keep in check. Any update or revision to any of these packages can cause cascading errors and issues down the line.



```
PS D:\nodejs\npm-demo> npm list
npm-demo@1.0.0 D:\nodejs\npm-demo
+-- express@4.17.1
| +-- accepts@1.3.7
| | +-- mime-types@2.1.27
| | | `-- mime-db@1.44.0
| | `-- negotiator@0.6.2
| +-- array-flatten@1.1.1
| +-- body-parser@1.19.0
| | +-- bytes@3.1.0
| | +-- content-type@1.0.4 deduped
| | +-- debug@2.6.9 deduped
| | +-- depd@1.1.2 deduped
| | +-- http-errors@1.7.2
| | | +-- depd@1.1.2 deduped
| | | +-- inherits@2.0.3
| | | +-- setprototypeof@1.1.1 dedupe
| | | +-- statuses@1.5.0 deduped
| | | `-- toidentifier@1.0.0
| | +-- iconv-lite@0.4.24
| | | `-- safer-buffer@2.1.2
| | +-- on-finished@2.3.0 deduped
| | +-- qs@6.7.0 deduped
```

What does our project serve to do?

BY Programmers FOR Programmers:

- Package management through update config. and version control
- Emailing System with customization (frequency and package selection)

Make it easier for programmers to manage
numerous GitHub projects

+

Ensure that packages are updated properly
to the user’s demand

Important Definitions:

Repository (REPO) - A repository is computer storage for maintaining data or software packages. This location contains files, databases, or information organized for quick access over a network or directly. A repo allows consolidating data with a version control system to store metadata for every file and log changes.

OAuth - An Authentication protocol that allows users to approve an application interacting with another on your behalf without giving away your password.

GIT - It is a version control system/tool used for file tracking and tracking changes in the source code.

Packages - A Package can be defined as a grouping of related types (classes, interfaces, enumerations and annotations) providing access protection and namespace management.

Tools Utilized in our project:

Front End will be a website built using SvelteKit and Bootstrap

SvelteKit - Framework providing project structure and JS-powered HTML

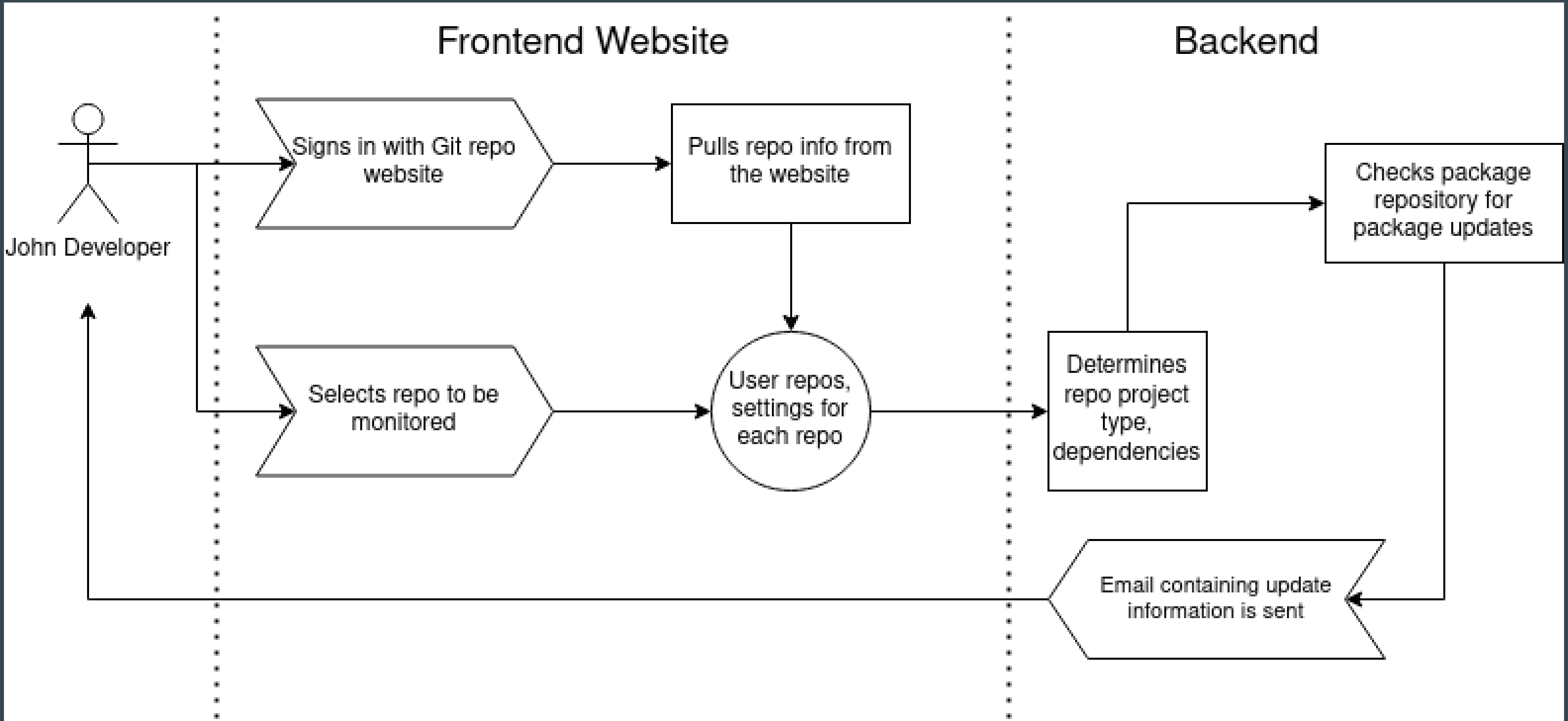
Bootstrap - CSS framework for JavaScript based web design

Back End will be built using .NET 7 in C#

.Net 7 - platform for robust back-end services with extensive libraries

Supabase will be utilized to help authenticate users in GitHub

Use-case Diagram


Frontend Website
Backend
John Developer
Signs in with Git repo website
Pulls repo info from the website
Selects repo to be monitored
User repos, settings for each repo
Determines repo project type, dependencies
Checks package repository for package updates
Email containing update information is sent

Future plans:

- Additional site logins (GitLab + BitBucket)
- Additional project types and package repositories (.NET, PIP, Yarn)

SRS Recap:

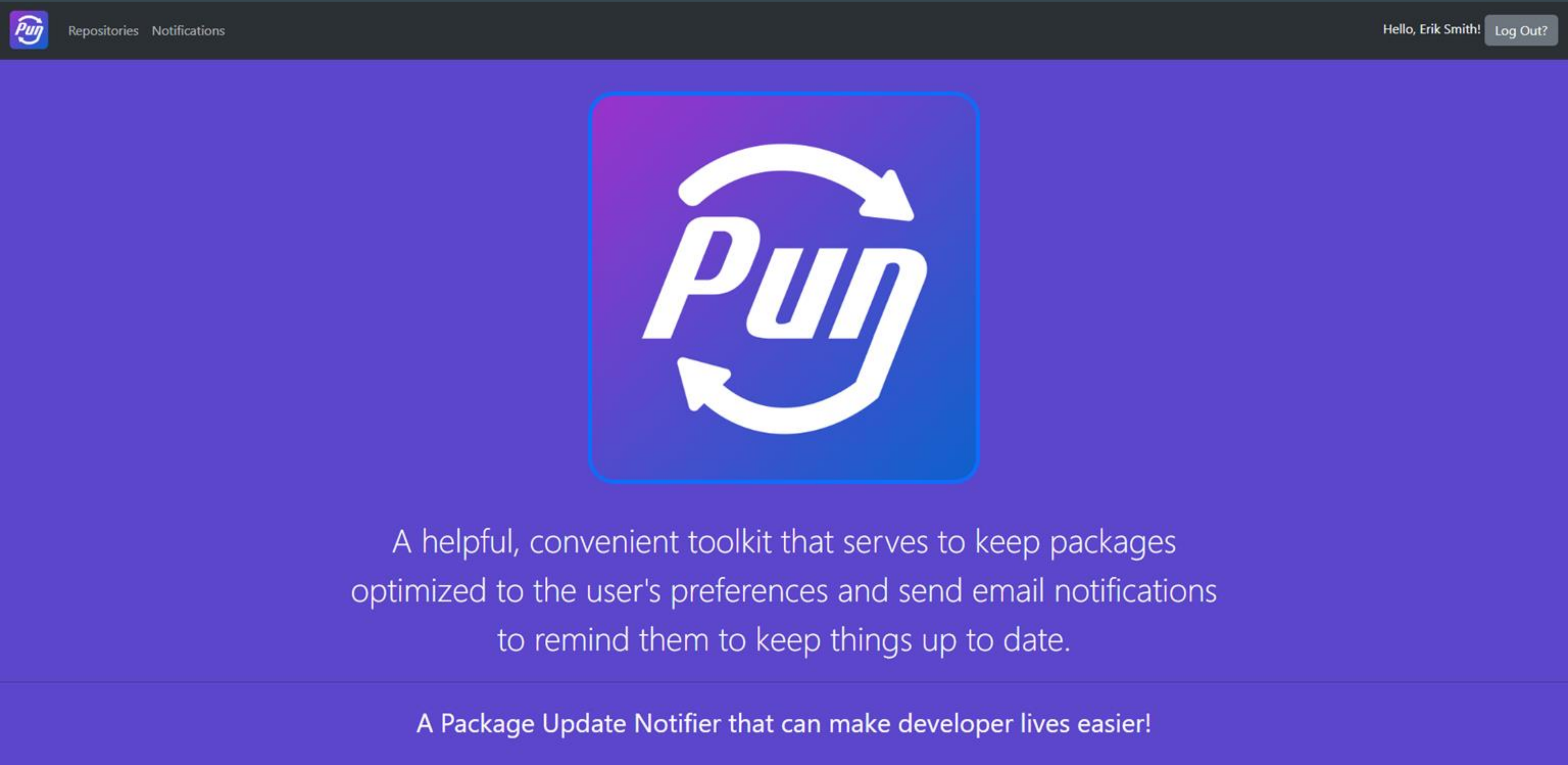
These are the following assumptions of our project:

1. The Client has a GITHUB account and is familiar with its application
2. The program should have access to the user’s repos and have the ability to read said repos
3. The program should have the ability to pull update information from package repos

Constraints

In order to sync in with the website they have to have Oauth integration which will allow read access to the users repositories. While this is the case with websites such as Github and Bitbucket already, this may not be the case for all websites, which may mean those websites would go unsupported.

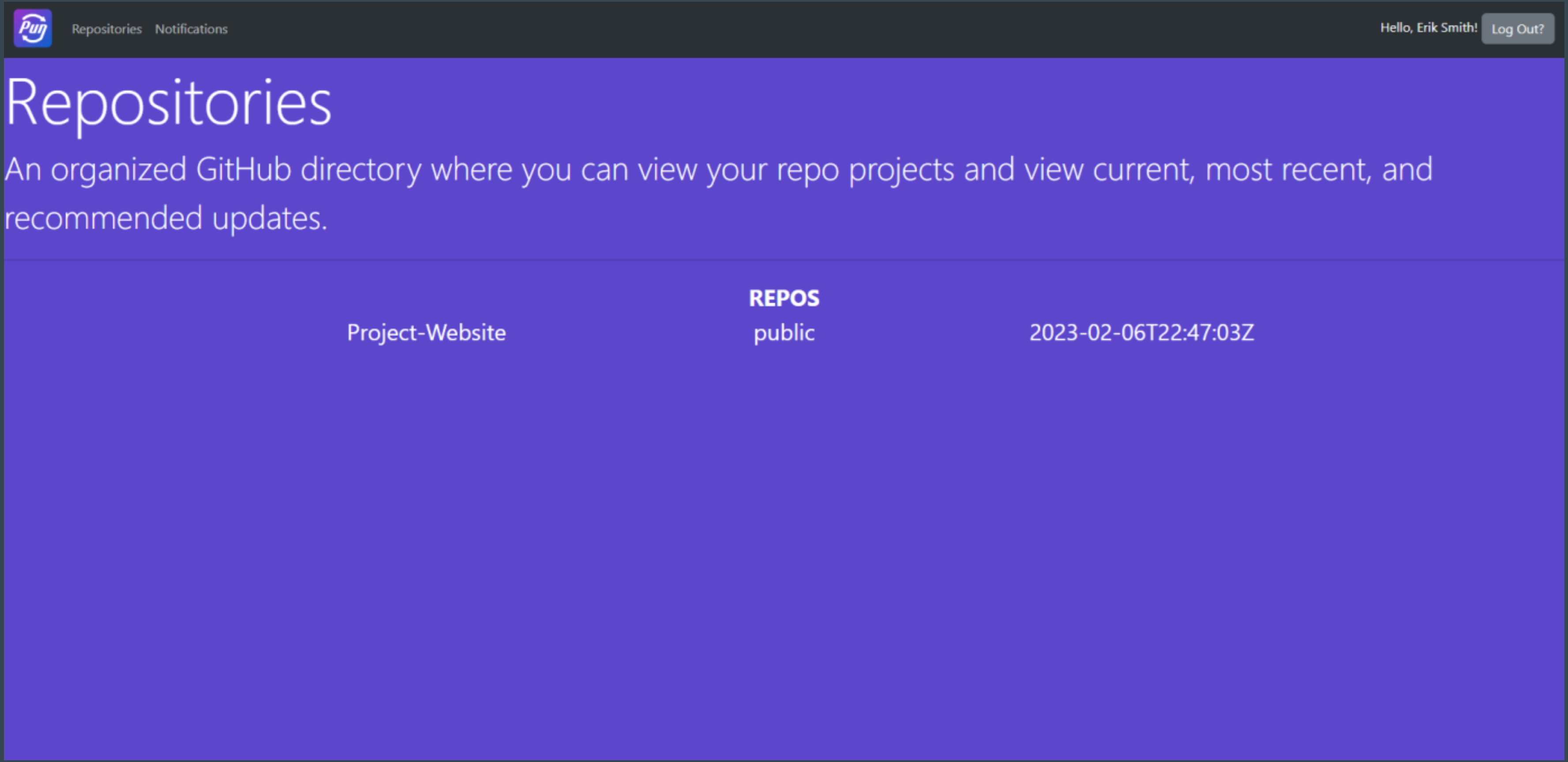
Front End UI



A helpful, convenient toolkit that serves to keep packages optimized to the user's preferences and send email notifications to remind them to keep things up to date.

A Package Update Notifier that can make developer lives easier!

Front End UI

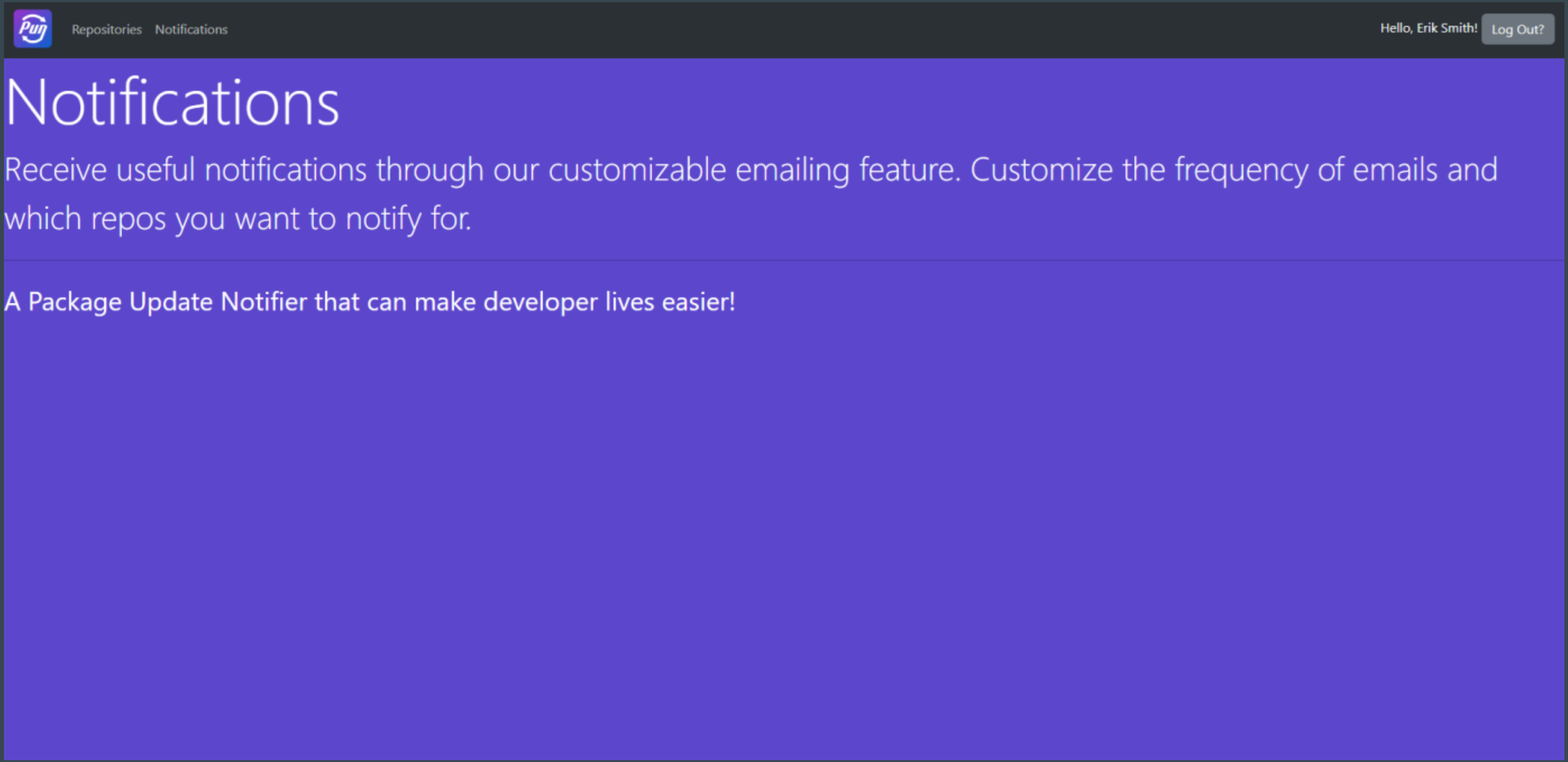


Repositories

An organized GitHub directory where you can view your repo projects and view current, most recent, and recommended updates.

	REPOS	
Project-Website	public	2023-02-06T22:47:03Z

Front End UI

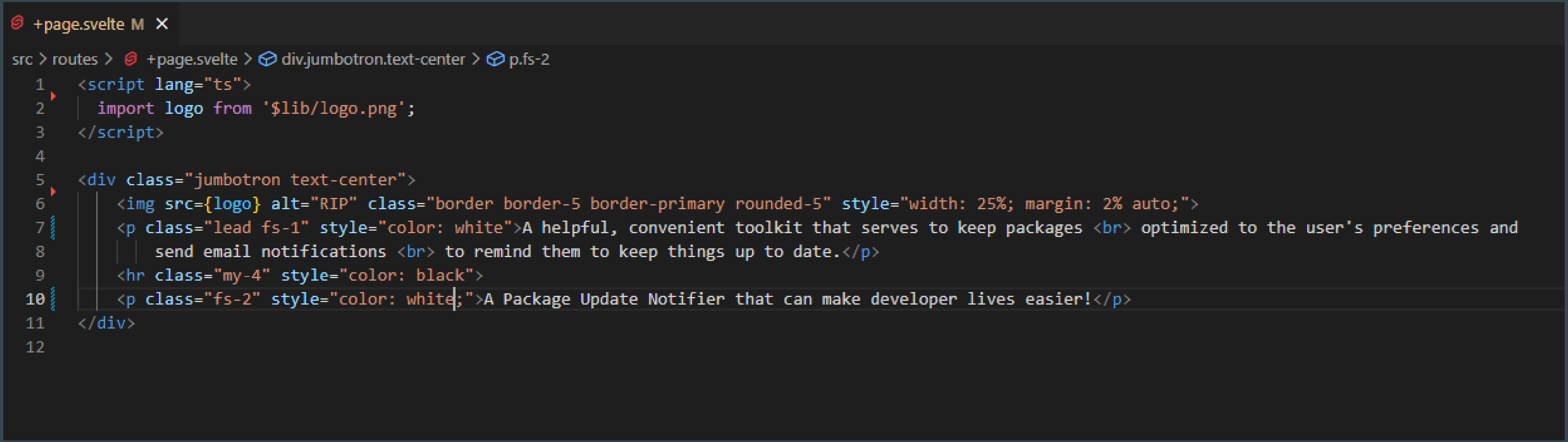


Notifications

Receive useful notifications through our customizable emailing feature. Customize the frequency of emails and which repos you want to notify for.

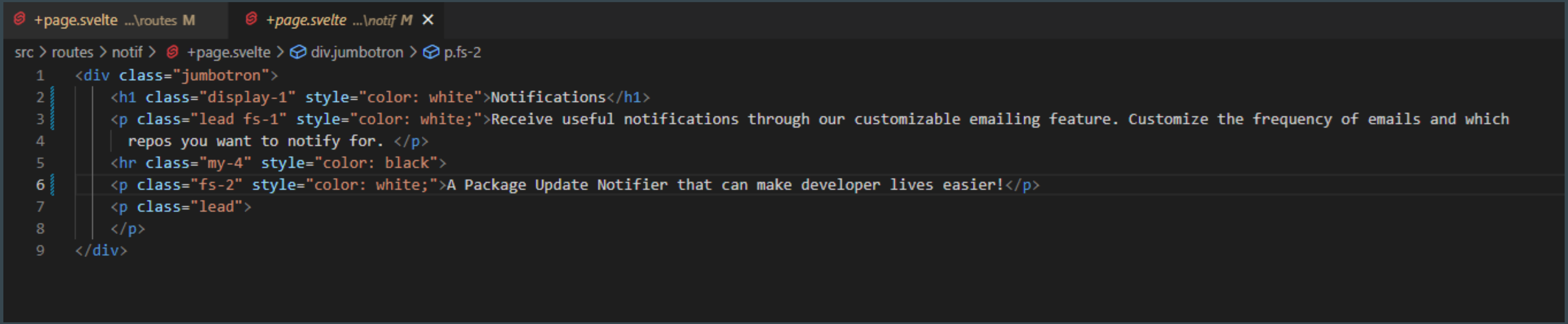
A Package Update Notifier that can make developer lives easier!

Snip bit of Front End Code


+page.svelte M ×
src > routes > +page.svelte > div.jumbotron.text-center > p.fs-2

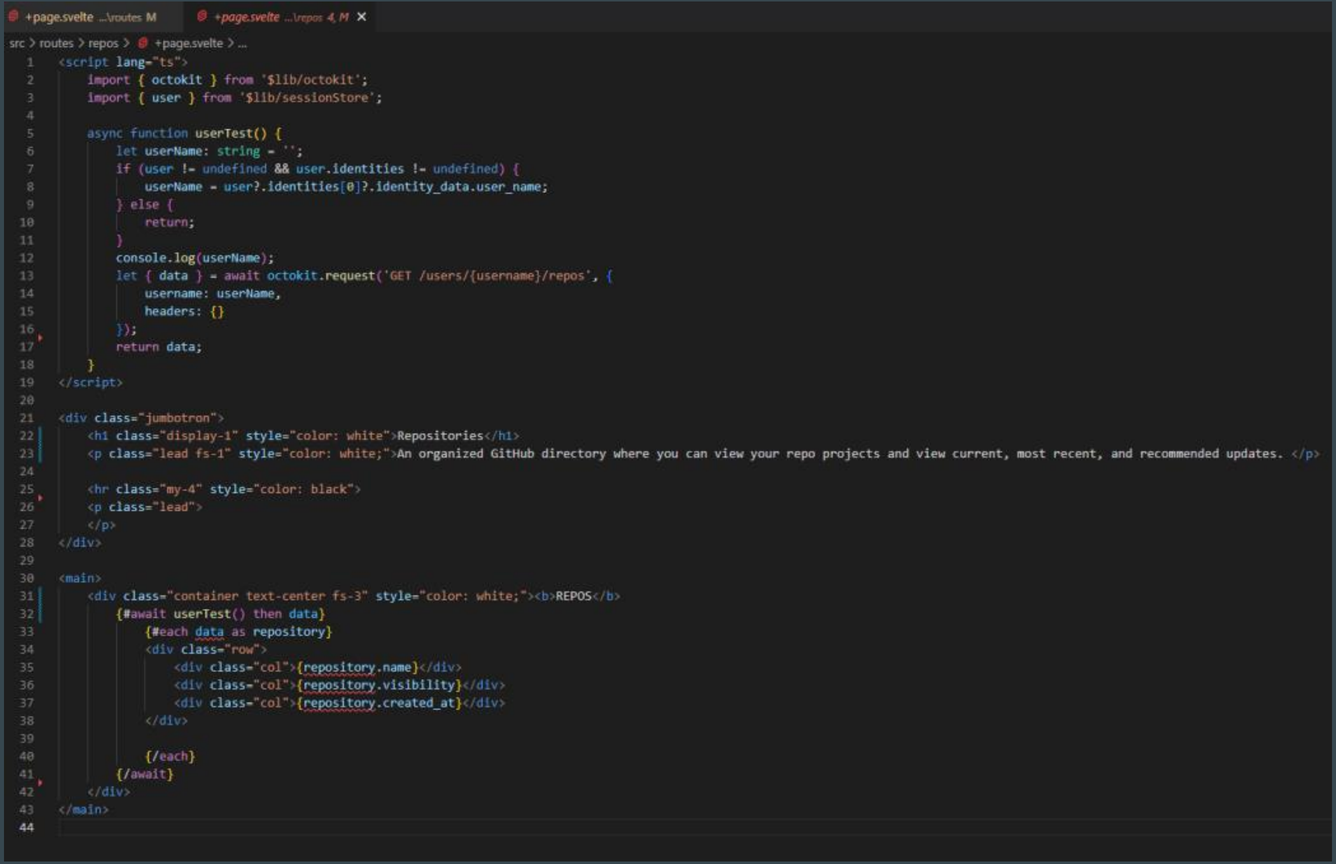
```
<script lang="ts">
  import logo from '$lib/logo.png';
</script>

<div class="jumbotron text-center">
    <img src={logo} alt="RIP" class="border border-5 border-primary rounded-5" style="width: 25%; margin: 2% auto;">
    <p class="lead fs-1" style="color: white">A helpful, convenient toolkit that serves to keep packages <br> optimized to the user's preferences and
        send email notifications <br> to remind them to keep things up to date.</p>
    <hr class="my-4" style="color: black">
    <p class="fs-2" style="color: white;">A Package Update Notifier that can make developer lives easier!</p>
</div>

```

Front End Code


+page.svelte ...\routes M
+page.svelte ...\notif M ×
src > routes > notif > +page.svelte > div.jumbotron > p.fs-2

```
<div class="jumbotron">
    <h1 class="display-1" style="color: white">Notifications</h1>
    <p class="lead fs-1" style="color: white;">Receive useful notifications through our customizable emailing feature. Customize the frequency of emails and which
      repos you want to notify for. </p>
    <hr class="my-4" style="color: black">
    <p class="fs-2" style="color: white;">A Package Update Notifier that can make developer lives easier!</p>
    <p class="lead">
    </p>
</div>
```

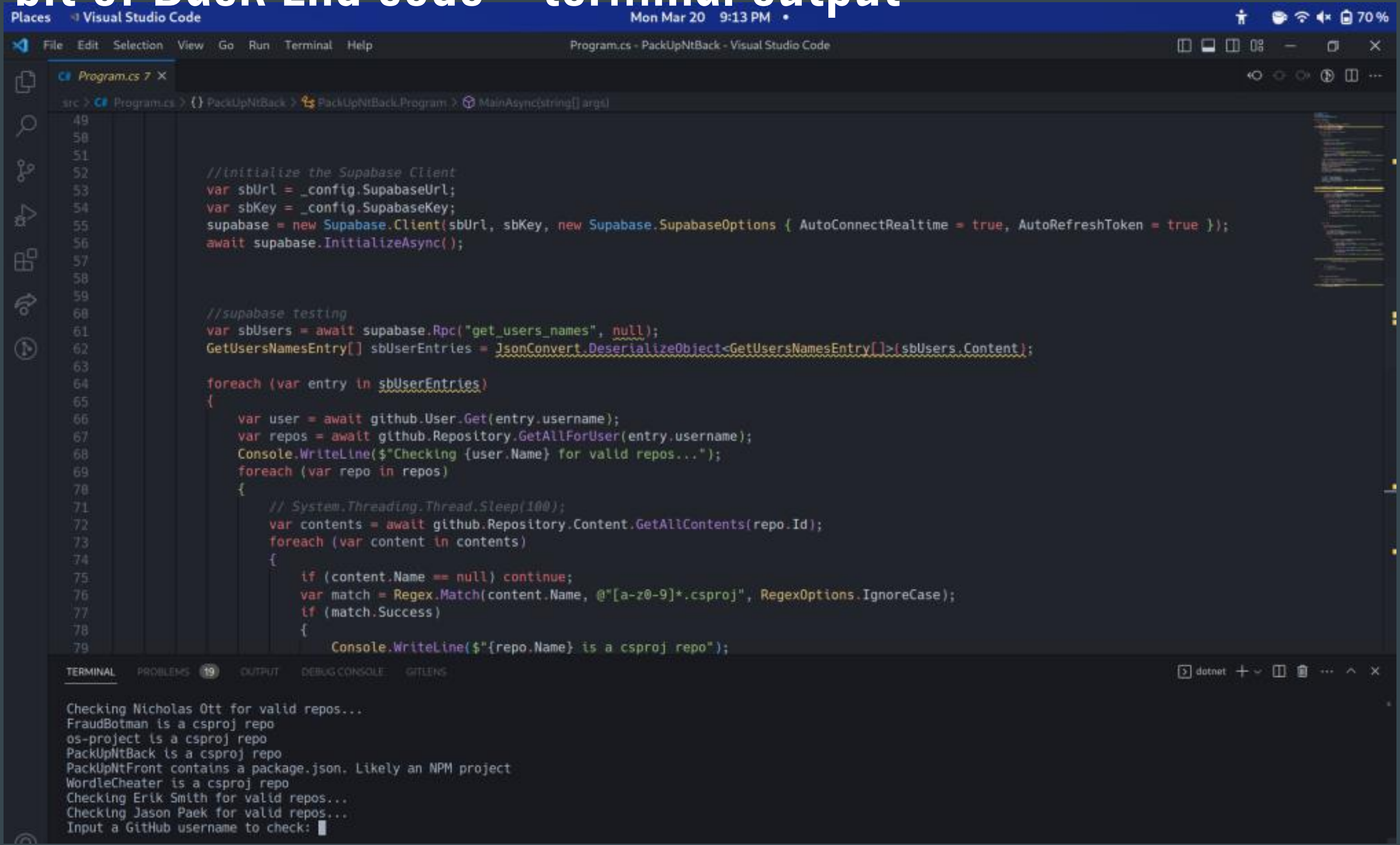
Front End Code



+page.svelte ...\routes M
+page.svelte ...\repos 4, M
src > routes > repos > +page.svelte > ...

```
<script lang="ts">
    import { octokit } from '$lib/octokit';
    import { user } from '$lib/sessionStore';

    async function userTest() {
        let userName: string = '';
        if (user != undefined && user.identities != undefined) {
            userName = user?.identities[0]?.identity_data.user_name;
        } else {
            return;
        }
        console.log(userName);
        let { data } = await octokit.request('GET /users/{username}/repos', {
            username: userName,
            headers: {}
        });
        return data;
    }
</script>

<div class="jumbotron">
    <h1 class="display-1" style="color: white">Repositories</h1>
    <p class="lead fs-1" style="color: white;">An organized GitHub directory where you can view your repo projects and view current, most recent, and recommended updates. </p>

    <hr class="my-4" style="color: black">
    <p class="lead">
    </p>
</div>

<main>
    <div class="container text-center fs-3" style="color: white;"><b>REPOS</b>
        {#await userTest() then data}
            {#each data as repository}
            <div class="row">
                <div class="col">{repository.name}</div>
                <div class="col">{repository.visibility}</div>
                <div class="col">{repository.created_at}</div>
            </div>

            {/each}
        {/await}
    </div>
</main>

```

Snip bit of Back End code + terminal output



```
//initialize the Supabase Client
var sbUrl = _config.SupabaseUrl;
var sbKey = _config.SupabaseKey;
supabase = new Supabase.Client(sbUrl, sbKey, new Supabase.SupabaseOptions { AutoConnectRealtime = true, AutoRefreshToken = true });
await supabase.InitializeAsync();



//supabase testing
var sbUsers = await supabase.Rpc("get_users_names", null);
GetUsersNamesEntry[] sbUserEntries = JsonConvert.DeserializeObject<GetUsersNamesEntry[]>(sbUsers.Content);

foreach (var entry in sbUserEntries)
{
    var user = await github.User.Get(entry.username);
    var repos = await github.Repository.GetAllForUser(entry.username);
    Console.WriteLine($"Checking {user.Name} for valid repos...");
    foreach (var repo in repos)
    {
        // System.Threading.Thread.Sleep(100);
        var contents = await github.Repository.Content.GetAllContents(repo.Id);
        foreach (var content in contents)
        {
            if (content.Name == null) continue;
            var match = Regex.Match(content.Name, @"[a-z0-9]*.csproj", RegexOptions.IgnoreCase);
            if (match.Success)
            {
                Console.WriteLine($"{repo.Name} is a csproj repo");
```

```
Checking Nicholas Ott for valid repos...
FraudBotman is a csproj repo
os-project is a csproj repo
PackUpNtBack is a csproj repo
PackUpNtFront contains a package.json. Likely an NPM project
WordleCheater is a csproj repo
Checking Erik Smith for valid repos...
Checking Jason Paek for valid repos...
Input a GitHub username to check:
```