MonkeyScripts.org
Employing modern learning techniques and collaborative filtering to improve user acquisition and retention in niche internet communities

Michael Aboff

Department of Information Systems and Technology, Wilmington University

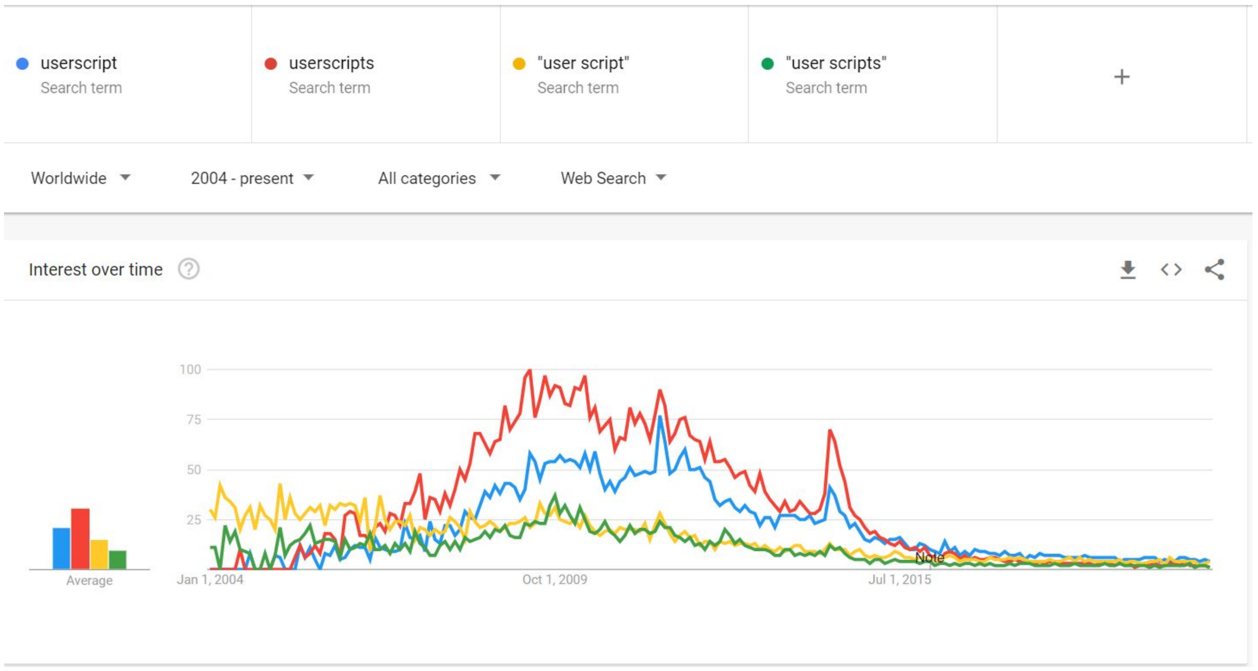
2020

Introduction

User scripts are small cross-browser add-ons that can be used to enhance a user's web browsing experience. These scripts are used to provide quick and unique customizations that directly modify the website a user is visiting. Created by the community, user scripts can add features ranging from adding detailed summary information from a review website into a favorite video streaming service to replacing all images with pictures of cats. (Pilgrim, 2005, pg 2)

While user scripts have no restriction as to what browser they can be implemented, they do require being installed into a user script manager. These managers are browser specific and perform the script execution when a given website is being loaded. The first user script manager began development in 2004 by Aaron Broodman (Pilgrim, 2009, pg. xiv) with its public release in 2005. Soon after, a script hosting site Userscripts.org was founded and became the major repository for all user scripts until it closed down in 2014 (Brinkmann, 2014). Newer managers and user script hosting sites have been developed since. As of August 2019, Tampermonkey (a modern manager) was the 13th most downloaded extension in the Google Chrome add-on store with over 10 million downloads (Extensionmonitor, 2019).

Tampermonkey's popularity has not corresponded with an overall increase in user script interest over the past decade. Compared to the first 5 years of rapid growth, Google searches for users have steadily decreased since 2011 as can be seen by this Google Trends report:

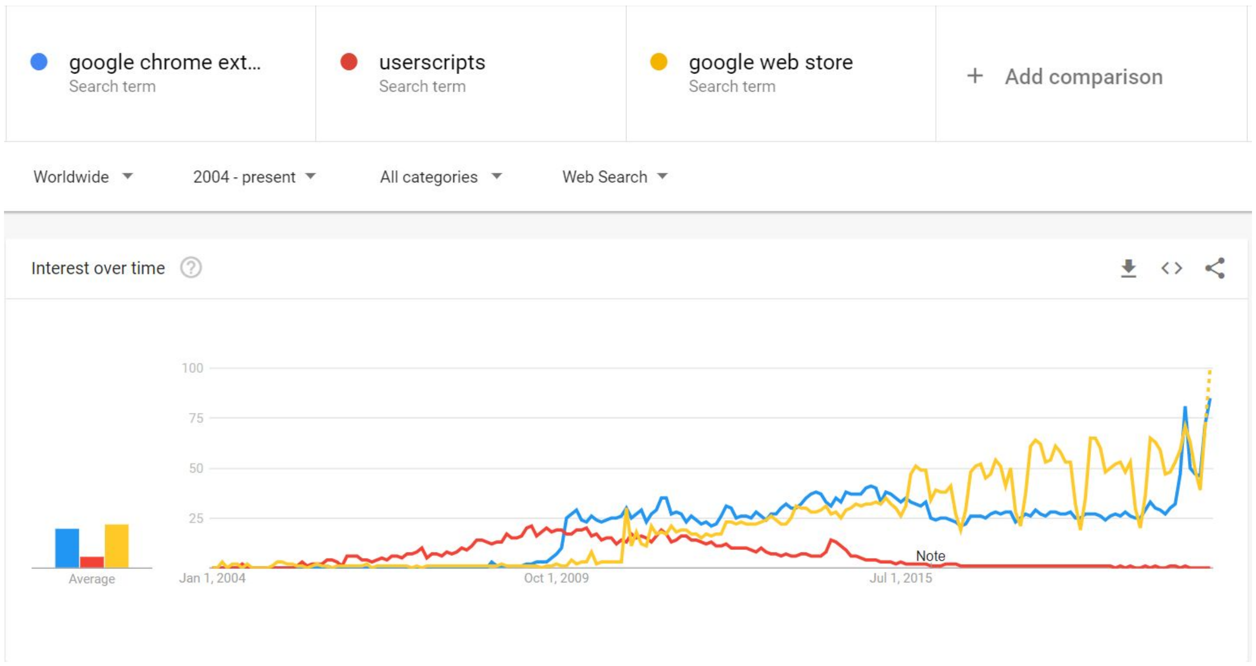


Data source: Google Trends (https://www.google.com/trends).

Searches for similar user script related resources (including popular managers and hosting sites) have followed similar interest trends, each with declining search terminology (see Appendix B, Fig. 1-5).

Statement of the Problem

There have been no published studies as to why userscript popularity has decreased since its peak in 2010, however one possible reason is Google's December 2009 release of extensions for their Chrome browser and, later, their own add-on repository called the Google Web Store. As the adoption of Chrome extensions increased there appears to be a decrease in interest in user scripts. While there is no empirical data determining causality, the Google Trends report appears to support this idea:



Data source: Google Trends (https://www.google.com/trends).

Extensions provide many of the same benefits of user scripts, they can change behavior of a website. Development of extensions has an initial learning curve and overhead that lets them work well with larger projects. Instead of being one single script, these add-ons contain multiple files and configurations. However, when a user wants to have many small modifications that affect many different sites, extensions become cumbersome to manage and develop. User scripts work well in these situations.

Increasing awareness and adoption of users scripts can help users eliminate frustrations they experience on websites they otherwise would have no recourse. This can lead to improved overall user satisfaction.

As there have been no formal attempts at determining why there has been a decline in popularity of user scripts, it is difficult to give any particular explanations. However, in this paper we will put forward the following several theories that we will examine and find lessons to be learned. Finally, by using the action research methodology of implementation and iteration, we will build a new user script repository website using the recommendations presented by this paper and launch it to the public.

Action Research Methodology

Niche and technical internet communities often struggle with increasing their user base and maintaining their interest. We posit that this is often due to a high bar of entry for new users to understand the topic and learn why it would benefit them. In order to improve interest we need to make the topic more accessible and relatable. To research and find a solution, I will be approaching this solution using an action research methodology which will enable immediate real world feedback instead of remaining in the realm of theory.

Action research is, as the name implies, an active process of building a tool or method to solve a problem, assess its effectiveness, and then iterate on the design as needed. Repeating this cycle allows for immediate feedback while being able to fine tune a product. This technique, also known as participatory research or collaborative inquiry, lends itself well for approaching our question by allowing us to test modern education and discovery techniques with a live audience. (McNiff, 2016)

This form of research contrasts with a more traditional form that navigates theory instead of creating a physical (or virtual) product. This traditional process, and its various methodologies, are accomplished by analyzing previous works as well as finding new information through surveys and interviews. While theory may help provide insight into what could be done, it is this author's belief that building and revisiting an actual product will provide faster and better-tuned results by actively receiving feedback and iterating on design. (Ulvik, 2018)

Focusing our research and development on the user script community. We will detail the journey of developing a website that introduces new users to this niche tool and attempts to lower the bar of entry by improving interest and relatability. Employing an action research methodology, we will build the website, receive feedback, reflect on it, and improve it.

To attempt to solve this problem, our website will approach the solution by implementing the following core features:

1. A concise and interactive tutorial that teaches how to use and find scripts.
2. A recommendation system that provides relatable and relevant results on a user-by-user basis.

Execution of this plan will be done in a series of three phases, spanning two weeks each. Below we will discuss each iteration, reflect on their results, and use the information to update the following stages.

Literature Review

Adult Education

Education is an essential component of making any technical tool for lowering the bar of entry to non-technical users. Expanding the user base for user scripts requires being able to explain a complicated topic down in an understandable way that allows the users of many backgrounds to feel motivated. In order to accomplish this, modern training and teaching techniques can be employed. The following review of literature explores the modern techniques and motivations of voluntary online learners by focusing on the growing field of adult education.

Wlodkowski and Ginsberg (2017) review the current understanding of adult education and how an educator can help motivate adults to learn. Older students, the authors explain, approach education differently than their younger counterparts. The book continues, instead of being required to attend mandatory courses, adults want to learn as it may provide them increased career opportunities, improve their social sphere, or simply build a new skill. Wlodkowski and Ginsberg explain that, as attendance is voluntary, it becomes even more important for educators to present material in a way that motivates the student to continue. This can be accomplished if the instructor follows the “Five Pillars,” which include, as the authors describe:

- Expertise. The instructor’s knowledge of the topic can inspire respect among the students and confidence in the teacher.
- Empathy. The understanding of the student’s expectations and goals for the course, and then modifying the lessons in response.
- Enthusiasm. The energy brought by the educator directly influences the student’s excitement of the material.
- Clarity. A clear and concise educational plan can prevent confusion and keep the learner interested.
- Cultural Responsiveness. Understanding the differences behind the student’s background and adjusting the material’s presentation to a way that can connect to broader society.

Even though Wlodkowski’s and Ginsberg’s book was designed primarily for in person education, the same rules can be generally applied to online education and training as well.

Understanding how adult students actually learn is essential for keeping them engaged and motivated through complicated lessons. In the first video of a three part instructional series, Rutigliano (2010) explains there are seven core principles that can explain what older learners expect from their education. The presenter explains each of these principles as such:

- Keep the lesson concise. Adult learners want to learn in the shortest route. These students are there by choice and want their education to be purposeful.
- Allow the student to be responsible for their own learning. Adult learners often wish to be directly engaged and active in their own education, providing them the space to do so can make a large difference in their engagement.

- Adults want to practice what they are taught. This helps reinforce the lessons and allow for quicker application.
- Older learners appreciate continual feedback. This lets them know how they are doing and, if needed, adjust course.
- Use rewards as motivation. As with continual feedback, rewarding encourages goal oriented adult students.
- The first and last pieces of information are the most important. This is due to them being the most likely to be remembered.
- There are three different learning styles that an educator should focus on: visual, auditory, and kinesthetic. Any lesson plan must be flexible enough to accommodate each.

Understanding the motivations of an adult student provides an essential piece of information to build any lesson plan off of and needs to be accommodated to keep the student motivated.

Being able to approach the three basic learning styles (visual, auditory, and kinesthetic) is essential when determining how to present to an audience. Rutigliano's (2010) second video goes into more depth as to how each of these types of people learn. Visual learners, the presenter explains, make up an estimated 65% of people. This visual style of learner, the video explains, retains information best when it is presented through text or video. 20% of learners, who fall in the auditory style, excel at picking up on vocal cues, such as cadence and pitch, that others may not notice, Rutigliano states. The final group are kinaesthetic learners who, the video explains, learn best by performing activities in a hands-on manner. By providing multiple educational styles to their students, an educator can keep more of their students engaged and motivated throughout the course.

Even if the motivations of a learner are understood by the professor, it is still necessary to understand what techniques can be used to do so. Products with complicated technical components can provide additional complications when presented to a user without a related background. In his 2018 book, Bixby explains that this can be largely due educators needing to cross cultural boundaries. Overcoming these obstacles can be done by focusing on five steps that can reduce the friction between cultures, as the book explains:

- Use easily understood graphics and pictures. Doing so can prevent alienation of students who may otherwise feel like a secondary audience instead of being actively engaged.
- Use regionally relevant case studies. When presenting examples, make sure it is relatable to the audience as it can help relate the lesson to their lives.
- Focus on imagery instead of text. Reducing unneeded complexity can reduce learner frustration and keep them motivated.
- Provide a detailed written student guide. Many students learn better in a written format or may need to simply revisit material at their own pace. A written guide can expand and cement the knowledge in the instructor's presentation.
- Obtain feedback from others in the target audience's culture first. This can allow the instructor to quickly optimize the material prior to the presentation leading to a better overall experience.

While Bixby presented these five steps in the context of regionally different cultures, the techniques can be generalized to any learning environment where a complicated, niche topic is taught to an audience who otherwise may struggle with it.

Focusing on internet e-learning environments, Kathleen King (2017) suggests the use of specific tools to keep students engaged. First, she proposes the use of a collaborative editing tool, such as a wiki where students can simultaneously work on the same document and work to build a communal knowledge base. Next, she recommends holding virtual consultation sessions and office hours where students can directly interact with the teacher. These meetings, the book says, can be performed in a variety of methods including video, voice, text, or virtual classroom. Third, the author states that designing projects with an emphasis of community engagement can help a student feel more engaged and connected to the assignment. King explains her fourth tool is prepared virtual solving which should be done by providing the students with vetted material they can use for their own research. The author's final recommendation is to encourage "collaborative knowledge development" by providing students with group projects where information can be learned and developed together. A focus on community building and user interaction is essential to keep adult learners engaged and motivated.

By focusing on the techniques used to understand and teach adult learners, the literature reveals common tools that can be applied in a variety of different situations. While this author was unable to find scholarly research specifically into building online tutorials for niche products, it is expected that lessons learned from the reviewed literature can apply for any online technical training.

Content Discovery

Educating users how to use a technical product, such as user scripts, is a necessary component in increasing the potential user base. Motivating them to explore the product and continue using it requires providing the user with content that is directly related to them. Without a convenient method of finding relevant scripts, interest in the tool will wane. In the following literature review, one can see that this requirement of relevant discovery is essential for modern products that may contain a lot of information.

One of the largest technology companies, Netflix, incorporate discovery algorithms into their core product and attribute much of their success to them. In their 2015 article, Netflix employees Gomez-Uribe and Hunt explain that users are “surprisingly bad at choosing between many options” and instead get overwhelmed choosing options that do not benefit them. The authors explain that their users will often lose interest after 60 to 90 seconds of searching, so it is essential that they are able to provide interesting results as quickly as possible. To solve this, the company described that they designed a series of recommendation algorithms that would populate the front page of their service. Gomez-Uribe and Hunt state that these different algorithms, many personalized by comparing the user’s preferences to others of similar users, are responsible for each different section of the front page. The authors explain that the personalized sections have been, and continue to be, essential for the Netflix business as it proves product worth to the user in seconds, lowering the risk of user disengagement. The developers conclude by explaining there were (at the time of writing) several open issues that a recommendation system inherently struggles with. Examples of these issues that were provided by the article include the cold-start problem for new members (which they attempted to mitigate by providing new users an initial survey of interests) and the need to mitigate run away positive feedback loops (when a set of recommended videos are engaged with more than others and become the only recommended). The authors conclude that their recommendation system is and will continue to be a pivotal role in their business' success.

Bobdilla et al. (2012) also explained that recommendations systems can be used to manage the large amount of data present in the modern internet. The article provides three general types of recommendation systems and explains them as such:

- Content-based filtering. This style recommended new items of similar topics of those used by and rated by the user.
- Demographic filtering. This technique provides recommendations based on the demographic information of the user, including age, gender, location, etc.
- Collaborative filtering. This style, commonly used by large software companies, attempts to find users who rate content similarly and provides recommendations enjoyed by these other users. At its simplest, what the authors call "memory-based", collaborative filtering ignores everything except using user ratings to find other similar users.

The author explained that there these recommendation filtering options can be hybridized with collaborative filtering in ways that can improve over the individual ones. The authors attempted improving basic collaborative filtering that used an algorithm called Jaccard Mean Squared Differences and found that a hybridized format did help

Carmel and Patt-Shamir (2016) explain that recommendation systems are integrated into large portions of society's lives from everything between selecting a movie to a romantic partner. The authors explain that, of recommendation systems, collaborative filtering is one of the most common styles used by the large scale technology companies such as Netflix to present users with relevant results. The article explains there are two types of collaborative filtering systems: interactive and non-interactive. The popular non-interactive recommendation systems, the researchers describe, are generated by taking all historical preference knowledge of users and passing it into an algorithm when an item is added and generating predicted preferences. It doesn't, the authors posit, account for changes after the fact and can fall into the "cold start" problem. Charmel and Patt-Shamir explain that interactive (aka competitive) algorithms can correct these problems by allowing the system to survey the user for their preference. The authors write that if you can ask the correct question you can limit the number of questions asked. Instead of simply providing a simple answer of whether or not the user prefers the product, the researchers found that having the user choose between two items leads to more accurate results. While the Carmel and Patt-Shamir explained they were able to properly build a model of user preference with a high probability of success, the worst case effort of determining preference was $\Omega(\lceil m^2/n \rceil)$ and they had not tested this in a real world situation. By implementing an interactive collaborative filtering algorithm, they were able to mitigate the cold start problem by providing side by side comparison surveys to users.

The literature shows that the use of a well designed recommendation system is essential for acquiring and retaining new users at the largest technical companies in the world. Use of such tools can improve user engagement and retention in smaller environments as well. Implementing an algorithm like memory-based collaborative filtering may be able to sufficiently motivate new users to join the user script community and improve overall interest.

Proposal

With this project we will be attempting to present a method of improving user acquisition costs and retention for niche and technical tools by focusing on one such tool, user scripts. Over the last decade, interest in user scripts have declined while other browser customization methods, such as extensions, have increased in popularity. For the purpose of this experiment, we will be focusing on two specific components that have been missed by all previous user script hosting solutions to date.

Understanding what user scripts are, their relevance, and how to use them has not been a focus of previous user script hostings sites. All previous sites have provided written explanations of user scripts and the steps needed to install them. As we learn from modern adult learning techniques in the literature reviewed, many learners require different presentation methods of information before they can easily understand it. In particular, we will create a tutorial that engages multiple learning methods, that should satisfy visual, auditory, and kinesthetic learners. Not only will we provide a detailed written guide, but also a video and an interactive tutorial that walks a user through the process of installing a user script.

After users learn what user scripts are, hosting sites must make it easy to find scripts relevant to their needs. Ease of discovery, a major business component for all virtual hosting companies such as Netflix and Amazon, has been traditionally neglected by all existing user script repositories. Existing solutions have provided a top downloaded and recently updated lists, as well as a search bar. In order to improve on this, we will implement a recommender system using a similar collaborative filtering algorithm employed at companies such as Netflix. By providing relevant recommendations and predicting customer preference, we believe we can improve user engagement and retention.

Over the next several weeks, we will be implementing the above features in a series of action research iterations. In the following sections I will explain what we plan to be involved in each stage.

Iteration 1

Our first stage will be to build the basic website structure. We will use modern backend and frontend frameworks to speed up development The anticipated milestones are as follows:

- Learn the Laravel back-end framework
- Learn the React front-end framework
- Create a wireframe design.
- Finalize color scheme and typography choices
- Use the wire frame design to start detailed design mockup
- Build the website backend with database integration.
- Build the website frontend using a Single Page Application framework and connecting it to the backend.

For a technical stack, I plan on using PHP/Laravel and MySQL as backend components. Javascript, React , HTML5, and CSS3 are the intended frontend components. It is essential that the design of the website is friendly, fast, and welcoming to new users. This will help the product feel more accessible to new users. During this phase I will use online tutorials and documentation to learn both Laravel and React. I will use Figma and Balsamiq to work on the design. No additional resources should be needed to accomplish these tasks.

Iteration 2

The second iteration will require development of core features that take advantage of the lessons learned in our literature review. We expect to accomplish the following milestones during this iteration phase:

- Implement user creation and authentication.
- Implement script submission, viewing, editing, and deletion.
- Research implementation techniques for collaborative filtering recommendation systems
- Implement a collaborative filtering recommendation system.
- Add a search functionality to the website.

This stage is critical for the success of the product as the tutorial and recommendation systems are essential distinguishing features for the website.

Iteration 3

The third stage will focus on preparing the website for final deployment. This includes implementing the design work into a usable website and building up the content for users to use right away. Anticipated milestones include:

- Implement the finalized web design into CSS.
- Develop additional user scripts to build an initial collection.

- Write a detailed tutorial of the installation process, limiting jargon to easily understood terminology.
- Record a video showing how to install user scripts.

With this completed, I should be ready to have users test and analyze how they interact with the site.

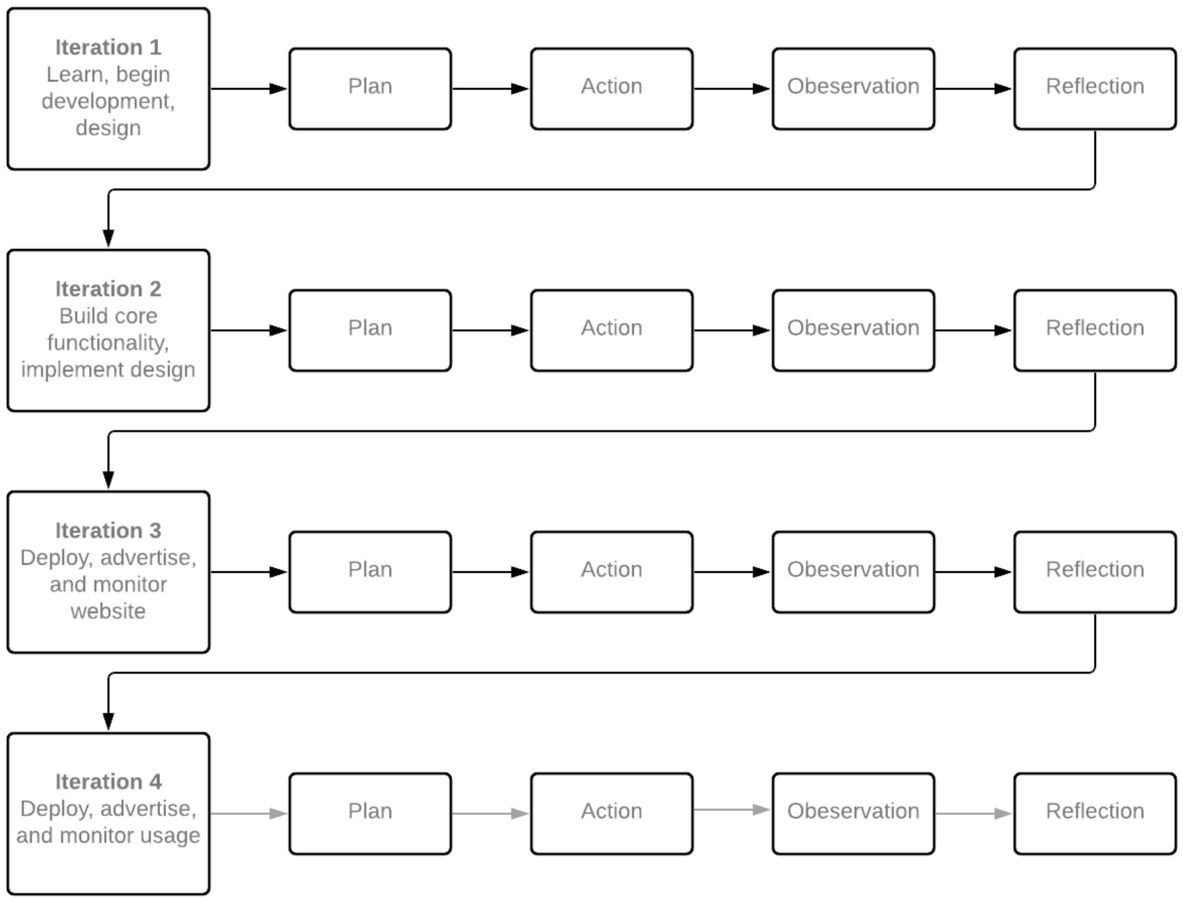
Iteration 4

The final stage will focus on publishing the website in a live environment and advertising the site on user script web forums. We will also wish to implement monitoring systems that anonymize data and allow us to receive immediate feedback on how users use the website while preserving their anonymity. Anticipated milestones include:

- Implement Google Analytics
- Implement an administration portal to allow for script and user moderation
- Publish website to the internet
- Advertise website
- Monitor anonymized user analytics

There are several forums and websites that allow for advertising of new projects that I will utilize to bring in new users and monitor user acquisition and retention.

The general iteration flow for this action research project can be illustrated by the below diagram:



Iteration 1
Learn, begin development, design
Plan
Action
Obeservation
Reflection
Iteration 2
Build core functionality, implement design
Plan
Action
Obeservation
Reflection
Iteration 3
Deploy, advertise, and monitor website
Plan
Action
Obeservation
Reflection
Iteration 4
Deploy, advertise, and monitor usage
Plan
Action
Obeservation
Reflection

Iteration 1

Plan

This first stage for developing anything is to first select the correct tools for the job. I decided I would use this project as an opportunity to learn a technology stack that lends itself to modern web applications. As such, the first of four iteration cycles would be focused on learning the tools and implementing an initial proof of concept.

I settled on using the following stack for development in following with modern techniques and ease of development. The backend would be developed using PHP 7, Laravel, and MySQL to provide a REST API server. Javascript, React, HTML5, CSS3, and Bootstrap would constitute the frontend of the website, following a Single Page Application architecture. To develop a design, a wireframe would be first developed with Etcher and the design finalized with Figma.

There are 2 main tasks that need to be accomplished for a successful first stage and to determine if any adjustments are required: 1) Wireframing; 2) Learning and Implementing both Laravel and React frameworks.

Wireframing

To build a website we must have an idea of what the final product contains. Based on personal experience, the way to accomplish this is build a wireframe that outlines the structure of the site, expected features, and general design. Though most user scripts are installed on desktops, to maintain a good mobile experience I will be developing both mobile and desktop designs, following a mobile-first mindset. I intend on creating the basic structure for the following basic pages and components:

- Home Page
  - Header with navigation links.
  - Banner message encouraging new users to learn more.
  - Multiple lists of recommended scripts
- Script Summary Page
  - Script metadata (summary, author, etc)
  - Script description
  - Install button
  - List of related scripts
- Script Code Page
  - Install button
  - Script code, formatted and syntax highlighted
  - List of related scripts
- Script Submission Page / Edit Page
  - Textboxes to allow for a script title, description, code, etc.
  - Save button
  - Delete button (edit page)
- Search/Result Page
  - A search bar
  - Filtering options (i.e. sort by rating, number of downloads, etc)
  - List of scripts

- Static Pages (Privacy Page, About Page, etc)
    - List of recommended scripts for users to start with.
    - Static text location
- Login/Registration
    - Username, password, email
    - Login button
    - Captcha (registration only)
    - Register button
- User Profile Page
    - User metadata (Join date, number of submitted scripts, average script rating, etc)
    - List of submitted scripts
- User Edit Page
    - Password edit section
    - Email edit section

These pages represent the core functionality of the site and necessary components for the final website.

Developing the wireframe will be done with the Balsamiq Wireframes tool, which allows for quick iteration of design. I expect this process to take 4 days and anticipate significant refining to be done during the formal design stage in Iteration 2.

Learning, Implementing Basic Laravel and React

Laravel is a popular PHP backend framework that enables fast and secure website development. Having decided early on that I wanted to use PHP 7 for this project Laravel was the MVC framework to choose. While I have used MVC frameworks and developed my own in the past, Laravel is a new tool for me so I would have to learn how to use it.

During this first iteration, I will be working on learning how to build a simple website with Laravel. In order to do so, I will follow online tutorials and documentation. During this phase, my goal will be to create a web page with users, scripts, and a home page that displays scripts in a variety of orders. I anticipate this phase to take 5 days before moving onto learning React.

React will be managing our front end Single Page Application. While I am experienced with Javascript, React is new for me. During this first iteration I plan to use online tutorials and documentation to learn this tool. I plan on using. After learning React, which I expect to take 3 days, I will attempt to build a home page, user login pages, as well as reaching out to the Laravel REST API to populate the page with information.

Action

Wireframing

I began the first iteration by building the wireframe for several of the core pages I listed previously. In particular, I focused on designing both the mobile and desktop versions of the Home, Search/Results, Script Summary, Script Code, and Static pages. Using Balsamiq, the process took 3 days.

Learning, Implementing Laravel and React

Despite my previous experience, learning Laravel was tricker than I originally anticipated. I took an online “Udemy” course over the course of 6 days during this first iteration. Following the program, I learned how to use the framework’s models, view, and controller structure, as well as the routing system and the built-in “artisan” commands. The artisan commands provide automated scaffolding and development tools (including a test server) that allows for quick programming. I set up a mysql server and connected it to Laravel, which allows for persistent storage of scripts, users, and more. I discovered and used the laravel/ui package in order to add a pre-built authentication system, which worked well for this prototype stage. By the end of the iteration phase, I built a basic website with user login and a home page that displayed previously stored scripts.

Over 4 days, I learned React using the official documentation. I learned how the framework breaks a website into different nodes called elements and components. The components will then only update the relevant portions of the page’s DOM, allowing for more efficient sites. Installation of React was taken care of by the laravel/ui package which installed essential front-end files. I was able to implement the home page by creating separate components for each section. Once the structure of the home page was created, I added a backend REST API endpoint in Laravel and set up the frontend to request the data using the Javascript fetch, async, and await methods.

Observation

Creating the wireframe as a first step ended up being an important step in determining what features I needed to focus first when developing the website. During the first iteration, I did not create a wireframe for every page listed as an important component in the planning section for this phase, but was able to create many of the core pages that can give a good sense of the style of design I am planning. Below is an example of the Home page wireframe, for all other wireframes please refer to Appendix C.

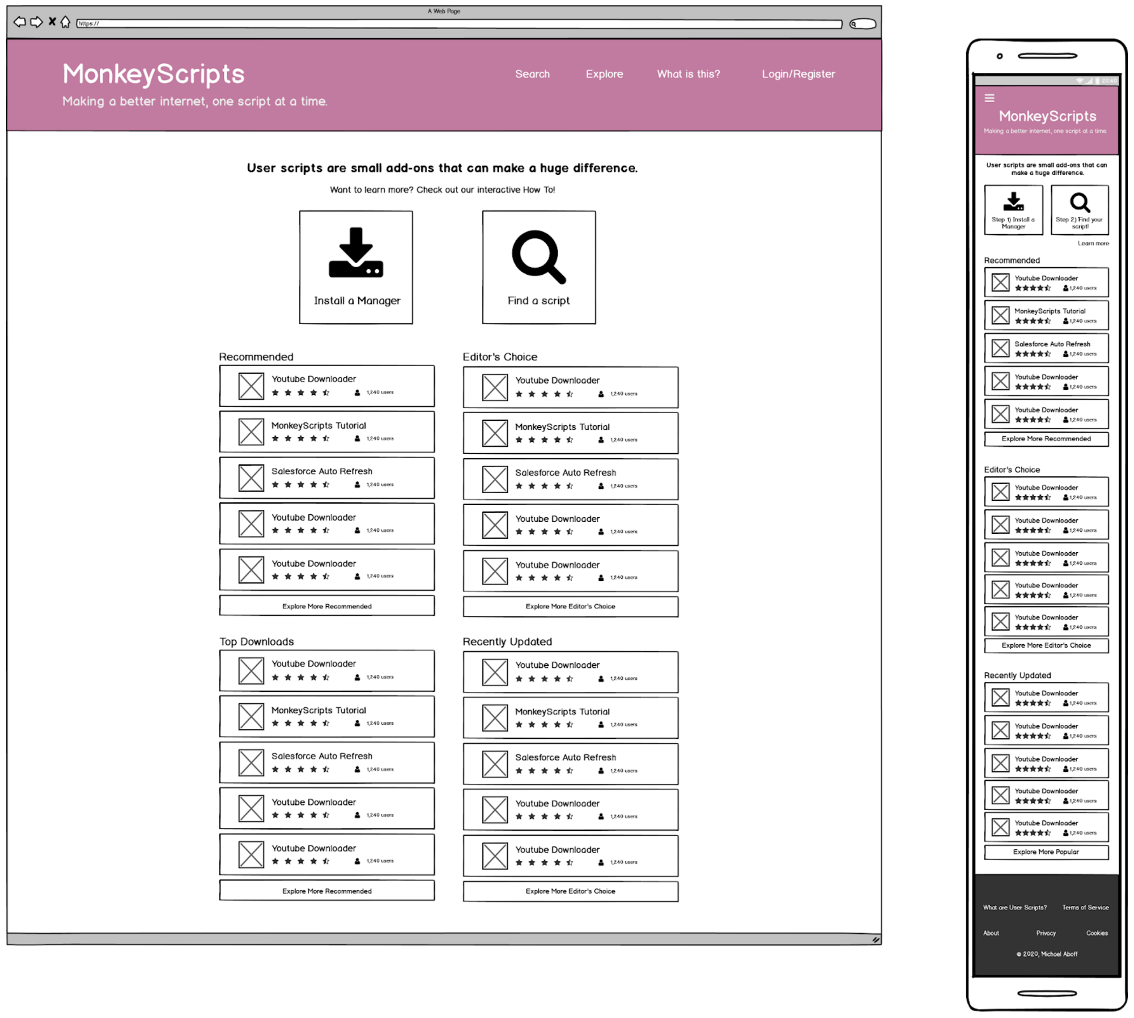
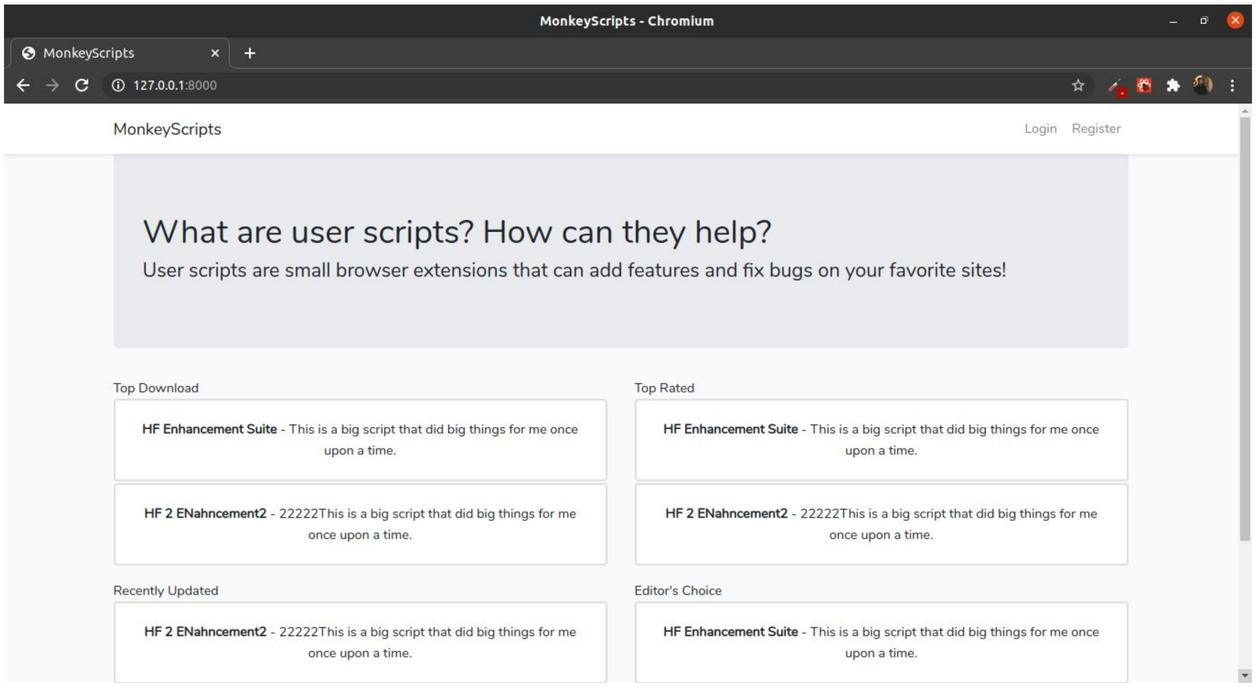


Figure C.1 - Home Page Wireframe

During this phase I was able to develop a basic website and REST API. The frontend is using REACT along with Javascript's built-in fetch method to request the information from the backend. I did not attempt to implement the design from the wireframe yet, and will not until a more formal design is created. Below you can see the current implementation of the design that demonstrates stored sample text in the cards containing each script.



What are user scripts? How can they help?

User scripts are small browser extensions that can add features and fix bugs on your favorite sites!

Top Download

HF Enhancement Suite - This is a big script that did big things for me once upon a time.

HF 2 ENahncement2 - 22222This is a big script that did big things for me once upon a time.

Top Rated

HF Enhancement Suite - This is a big script that did big things for me once upon a time.

HF 2 ENahncement2 - 22222This is a big script that did big things for me once upon a time.

Recently Updated

HF 2 ENahncement2 - 22222This is a big script that did big things for me once upon a time.

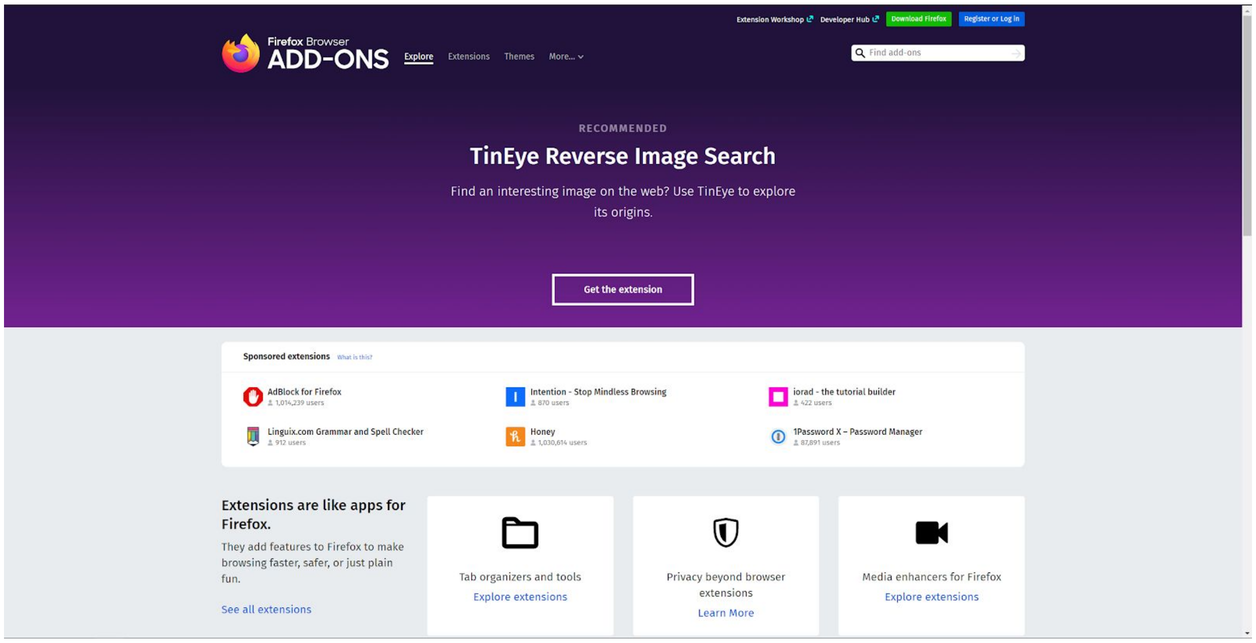
Editor's Choice

HF Enhancement Suite - This is a big script that did big things for me once upon a time.

Figure: First Iteration of Home Page

Reflection

As I designed the wireframe, it did a very good job helping me understand what features should be required for the website. For example, in order to encourage users to learn more about user scripts, I realized how important it is to use the space at the front of the fold. Reviewing Firefox's Add-On Extension store, one can see that approximately a quarter of the "above the fold" space is used to explain what extensions are in a friendly and welcoming manner:



TinEye Reverse Image Search

Find an interesting image on the web? Use TinEye to explore its origins.

AdBlock for Firefox 1,014,239 users	Intention - Stop Mindless Browsing 870 users	iorad - the tutorial builder 422 users
Linguix.com Grammar and Spell Checker 912 users	Honey 1,030,614 users	1Password X – Password Manager 87,891 users

Extensions are like apps for Firefox.

They add features to Firefox to make browsing faster, safer, or just plain fun.

Tab organizers and tools

Privacy beyond browser extensions

Media enhancers for Firefox

While creating MonkeyScript's wireframes, it became very clear that we needed to have something similar. On each page, a banner that quickly explains what user scripts are and encourages the visitor to learn more about how to use them. These banners will link directly to an interactive tutorial.

What did not go well, however, were the particulars of the designs themselves. While going through each page, my ideas changed and I did not do a good job going back to maintain consistency. This is clear when looking at the size of the content container. The content for the home page wireframe (Figure C.1) is considerably skinnier than the script content page (Figure C.2) which leads to an inconsistent design. Due to time constraints, I had made the decision to leave the wireframes as they are and focus on creating a more consistent and finalized design prototype during Iteration 2. The primary constraint on this task was time.

Learning Laravel and React has been a long term goal of mine, as they are both heavily used in the industry. Creating the basic structure of the website is an essential first step that will enable rapid development of other features in following iteration phases.

The original plan was that user authentication and registration would be completed by the end of this stage. Unfortunately the process of learning the technologies took longer than I had expected it to, and these features had to be moved to Iteration 2.

While I was able to implement user authentication using Laravel's built in frontend templating system (blade), this would not work for a REST API based frontend like we would be using with React. Moving forward, I will be removing the laravel/ui authentication scaffolding that was put in place and implementing the official Laravel Passport package. Passport will provide OAuth2. This should provide the ability to interface with other login services, such as Google's and Github's, which can lead to a more user friendly experience.

I was also unable to create script submission, edit, or viewing pages during the first two weeks of the project. These features will have to be moved to the second phase.

Now that the main training period has completed, I expect continued development to progress quickly.

Iteration 2

Plan

The second iteration covered the next two weeks (third and fourth weeks) of the project. The originally determined plan for this iteration was to jump off of the self-education completed in the first iteration and implement core features into the product. I hoped to complete the following tasks for this week:

- Implement user creation and authentication.
- Implement script submission, viewing, editing, and deletion.
- Research implementation techniques for collaborative filtering recommendation systems
- Implement a collaborative filtering recommendation system.
- Add a search functionality to the website.

To implement these features, there was a requirement that I had a strong understanding of the technologies at use. I did not anticipate the length of time it would take to learn the new technology stack, which led to several required actions to have to be moved to this iteration. As such, I needed to change the plan for this cycle.

Implement the front-end as a Single Page Application

Single Page Applications are a popular and modern design pattern where the majority of the page will be loaded when first accessing the website. This has the benefit of making the website feel overall responsive and quick as the site is used, as it only loads the specific data it needs while you navigate the site. This is a core reason for the use of React in this project. During the first iteration, I found out that React did not have a built-in way of managing multiple page routes. For these two weeks, I would need to research and find out how other websites manage to use React as an SPA with multiple pages.

Design prototype, typography, and color theme.

I was able to make good progress during the first iteration in developing a wireframe, but was unable to start the actual design or determine the color theme/typography. Before starting, I need to perform more research into good typography and color theory to try to find a color theme that works with what I am trying to accomplish. I hope to be able to start working on the actual design prototype

Add User Authentication

Users need to be able to log into this site so we can properly associate the user scripts. Users should be able to log in, create scripts, and eventually have customized recommendations. Without this functionality,

Add Script functionality

Another core functionality of this website is the ability to submit, view, edit, and install user scripts. Throughout the next two weeks, I plan on building the structure for this so we can have scripts that are ready to search for, and be recommended.

Add Search Functionality to the website

The reason behind this website is to make the process of user script discovery easy. Every part of the design and every feature implemented needs to incorporate this into its DNA. As such, implementing a Search tool is necessary. While it is not one of the focuses of this research project, I will plan on researching 3rd party alternatives to provide a quick solution.

Begin research and work on a recommendation system

During my literature review, we learned about the importance of recommendation systems. The actual implementation of them, however, is a different story. During this iteration, I need to make progress on implementation of the recommendation system. Successful implementation will hopefully be able to distinguish between this project and the current established competition.

Action

Design

The first two days of this iteration, I started with research into color theory and typography theory to attempt to determine what themes I would want to choose. I read several articles and viewed many color swatches on color inspiration websites. I then began attempting to implement them into a design. To decide on the website's typography, I also performed research, investigating the I found that colors portray different emotions. For example, here are some of the basic color and feeling connections (Wong, 2020) (Chapman, 2010) (Cao, 2015):

- **Green:** Wealth, growth.
- **Blue:** Calmness, safety.
- **Purple:** Luxuriousness, romance, royalty.
- **Brown:** Dependability, reliability.
- **Orange:** Friendliness, enthusiasm

- **Red:** Passion, urgency, danger
- **Yellow:** Youthfulness, cheerfulness.
- **Gray:** Futurism, logical.
- **White:** Cleanliness, virtuism.
- **Black:** Elegance, power, sophistication.

I see this project's mission as attempting to make a complicated, intimidating, and technical topic easily accessible and understandable. I need every part of the website to work towards this goal, and this includes the use of colors. Following the above color mappings, I determined the core colors would either be a shade of blue or brown. I couldn't decide at this time and knew that I would have to do more research.

Finding my choice in typography was considerably more easy. Navigating through online font repositories I determined quickly that I would want to use Titan One for my logo font and Roboto for the rest of the site. The two paired very well, leading to a youthful and energetic feeling, while also being easy to read and professional.

Single Page Application

To implement a single page application, I spent the next three days days researching and implementing a plugin called React Router. React already works by selectively rendering only certain portions of the page at a time, the router plugin just provides easier access to making conditional statements about what components to render based on what URL the browser is pointed to. I was able to use this to start the structure of the single page app.

User Authentication

Before I could start working on properly implementing scripts I would need to add basic user authorization. I first looked into the laravel/ui authentication scaffolding, which my original research had pointed me in the direction. After implementing it, I realized that it did not have the token based authentication I would need in order to provide login access for the separated React front-end I was developing. From there I switched to using Laravel Passport, a fully fledged oauth2 server implementation which could be connected to an API and provide tokens that can be stored in the front end. Going through both of these technologies took 3 days to complete.

Front-End Interfaces

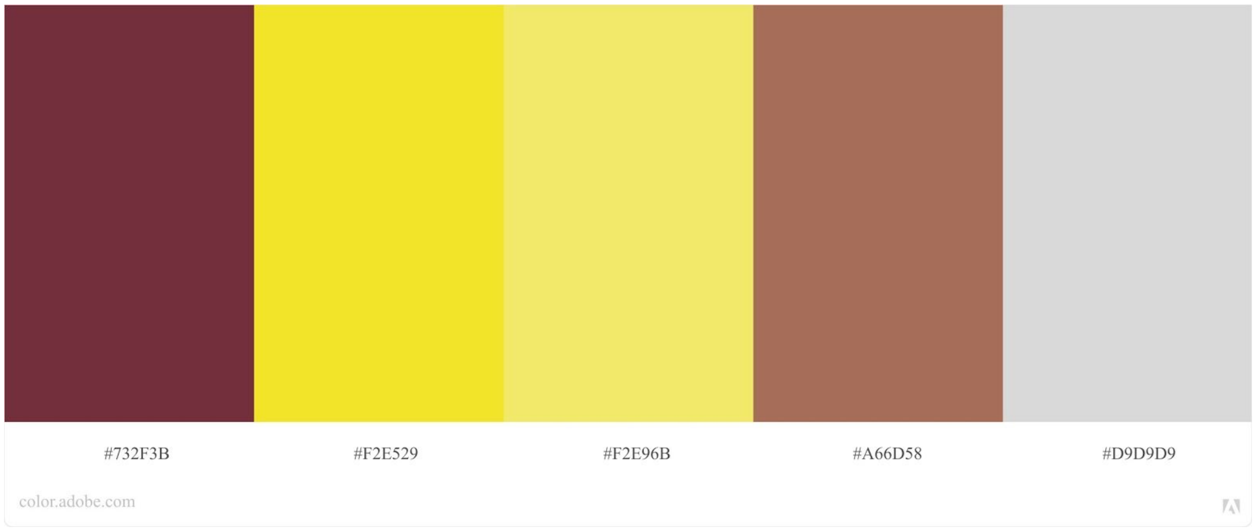
To simplify both the code and troubleshooting, I decided to split many of the common tasks into separate Interfaces on the front-end. This helps keep React components single-issue as well, making them easier to program. If I decide to ever change the libraries to interact with external components (i.e. using ajax instead of fetch for networking requests) I would only have to modify one location, as long as I leave the interface's externally facing commands the same.

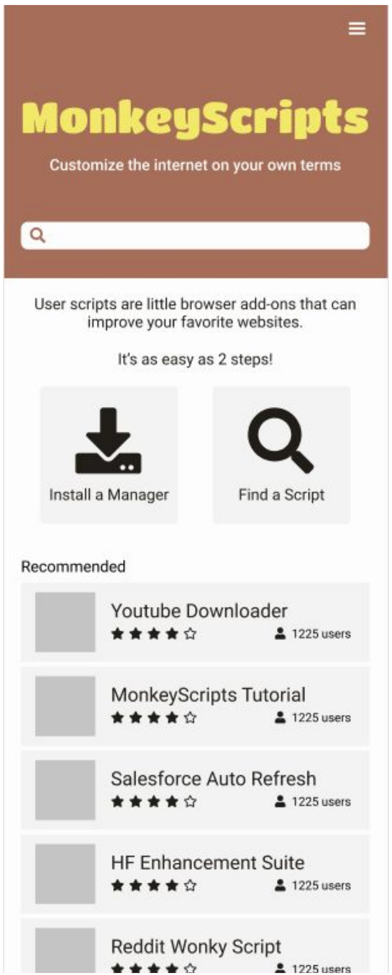
While not a part of the original plan, I spent less than a day on this and it would provide immediate time savings.

Observations

Determining brand identity is no easy task. Immediately beginning this project I knew a few things. The website would be called MonkeyScripts and that every decision should be made with a focus on making the user feel comfortable and motivated to learn about user scripts. I did not anticipate, however, how difficult it would be to decide on a final color theme for the site.

Originally, I wanted to use this brown color theme. I believed it provided a homey feel, as described in many of the color theory blogs and articles I read. Unfortunately, after implementing it into a design prototype, it became clear that simply following theory "rules" was not going to be sufficient. Immediate feedback from friends and family encouraged me to quickly drop it as an option, I agree wholeheartedly with their opinions.





User scripts are little browser add-ons that can improve your favorite websites.

It's as easy as 2 steps!

Youtube Downloader ★★★★☆	1225 users
MonkeyScripts Tutorial ★★★★☆	1225 users
Salesforce Auto Refresh ★★★★☆	1225 users
HF Enhancement Suite ★★★★☆	1225 users
Reddit Wonky Script ★★★★☆	1225 users

The typography was much easier to decide on. I have used Roboto in the past and appreciate it's youthful professionalism that does not come off as intimidating. For the logo font, I knew I wanted to have a playful font that would fit well with the main Roboto font. Browsing through Google Fonts, I found Titan One.

MonkeyScripts

Customize the internet on your own terms

My focus for this week was to learn and implement API/cookie based authentication. I had previously been using backend-managed, session based authentication, which worked perfectly fine and had a pre-built framework included in the laravel/ui package I was already using. However, as my goal is to have a front-end that is separate from the backend, this would not work. The front-end client needs to be able to request information on behalf of the user themselves, in order to do this it needs to be able to be able to submit a login request to the backend and store a valid authentication key. To do this, I relied on Laravel's recommended OAuth2 package called Passport.

I was able to implement this tool by setting up React to use the javascript fetch api to send a username and password to a backend API endpoint. Passport would then respond with a JSON response that includes an authentication token, refresh token, and expiry time. The refresh token can be used to bump the expiry time for users. While this seems like it may be simple, there was a bit more complexity in the procedure. Passport requires that additional parameters be sent as well, including a secret client key that should never actually be shared with any user. To get around this issue, I created something called a “middleware”. This is a script that can be set to run and modify the request before the controller ever receives it, but after the router tries to determine how to manage it. This filled in the remaining information and the request was finally passed to Passport which would generate the tokens.

To store this token in the frontend, I used the universal-cookie package for javascript. This provides easy access to get, set, and remove cookies in the browser.

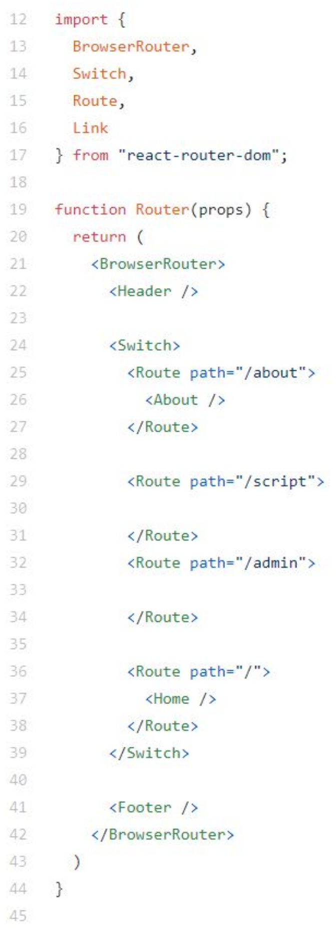
Passport does not have the ability to register any users, so I had to create a separate Controller to accommodate. This new controller accepts a username, email address, and password and then generates a new user account.

Users can now log in and authenticate via the GUI. Next steps are to make log-ins affect the state of the front-end application and to limit individual pages from being seen if you are not logged in (this has to be done on the frontend and backend).

To accommodate all of the steps needed to implement token based authentication, I needed to implement basic interfaces that could be called simply without having to rewrite a lot of code. This was a quick and very successful choice that would help me throughout the rest of this iteration and almost certainly would into the rest of the project.

With user authentication and creation, I found that I also needed a way to properly validate that user information was being submitted properly. I found that Laravel has validation tool that helps define easy rules for user implementation. I implemented this to immediate success. Unfortunately a major problem with this tool is that it would immediately return an error message to the user, which isn't ideal when I'm trying to have full control over the info sent to the front-end. I was unable to find a way to override the error without overwriting the backend code itself. In the end I used a combination of the validator tool and some conditional statements to check for common issues to limit the number of errors that might reach the end user.

The implementation of a React single page app was entirely feasible due to React Router. This tool let me easily define a single place which could overload the content of the page to whatever URL the user was directed to. The Router.php page was as simple as this:



```
import {
  BrowserRouter,
  Switch,
  Route,
  Link
} from "react-router-dom";

function Router(props) {
  return (
    <BrowserRouter>
      <Header />

      <Switch>
        <Route path="/about">
          <About />
        </Route>

        <Route path="/script">

        </Route>
        <Route path="/admin">

        </Route>

        <Route path="/">
          <Home />
        </Route>
      </Switch>

      <Footer />
    </BrowserRouter>
  )
}

```

Reflection

I was able to make some essential progress during these two weeks. I was unable to make as much progress as I needed to for this week. Largely this was due to additional research being needed to better understand the core technologies while implementing user authentication and converting the front-end to a proper Single Page Application. I did not anticipate how much more work would be needed to learn two entire frameworks while also managing a personal and professional life. I was unable to get to implementing scripts, search, or the recommendation system. I will have to make this up during the upcoming third iteration.

Iteration 3

Plan

The third iteration covers weeks five and six of the project. Originally the plan for this iteration was to implement the final design, add general content, and polish the site to be ready for release in weeks seven and eight. As discussed in the previous section, core site functionality still needs to be added and the schedule needs to be updated. During these two weeks I plan on accomplishing the following to catch up and release the website during Iteration 4:

- Implement script submission, viewing, editing, installation, and deletion
- Restrict editing and deletion to only be available for signed in users
- Implement search
- Implement profile pages
- Implement the interactive tutorial
- Research and implement the recommender system
- Determine the color theme for the website and start development of the design

Content creation and other requirements will likely need to be pushed to the fourth iteration so we can get a released website and start getting user feedback.

Actions

Script Management

Without being able to submit user scripts, it would be difficult to host a script sharing website! I spent the first few days working on building the first React component heavy page on the site, a script submission page. I was able to build it in such a way that the same code could be used for editing an already submitted script, which will save me a large amount of time while implementing the design. This part of the process took me about 3 days to become more familiar with how React components work, connect it to the backend for script storage, display it, and edit it.

Script Viewing and Installation

After being able to submit a script, it is natural to add a way to view and verify the script. This was also a natural time to implement script installation. I spent the fourth day working on getting this working as well as starting to add a basic script list to the front page.

Tutorial

One of the key innovations I am implementing for this project is to create an interactive tutorial, something that has not been done on any browser extension or user script repository website. The action for this iteration is to develop the interactive component, a script that the user can install and get immediate feedback to let them know that they were successful. I developed a user script and the basic tutorial page (with filler text). The script changes a red box into a green box when activated. This took me one evening to complete.

Search

In line with the core focus on lowering the bar for content discovery, adding search was essential for this to work. To save time, I implemented a 3rd party indexing and search tool called Algolia by using Laravel's built in Scout interface library. This process was quick to implement and took a day to implement.

Profile Pages

When a user accesses the site, they may want to explore different scripts submitted by their favorite creators. As such, I created a user profile page that lists basic detailed information for each user as well as providing a list of all the scripts they have created to date. This took about half a day to implement.

Contexts

React components essentially operate independently from each other and will only update if they detect there is a change to something they are displaying. In order to get one component (the login code) to tell other components (i.e. the navigation bar) that something has changed, we can pass this information via a tool called contexts. Implementing this let me inform the rest of the site that the user was logged in.

User Interactions

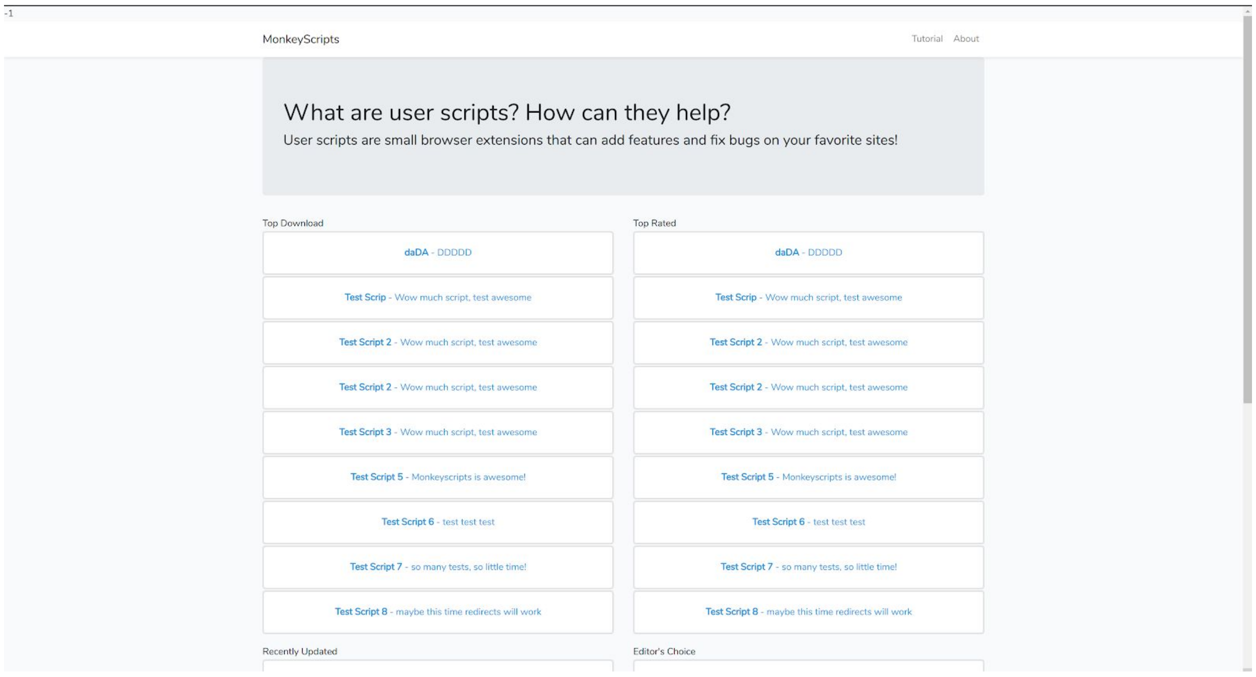
In order to build a collaborative filter recommendation system, I had to record user interest. Normally this is done with a rating system (i.e. a 1 to 5 star rating), but I determined I didn't have time to implement something like this. Instead I decided to determine user interest by recording if they viewed a script and if they decided to download it. Downloading would be a higher score. I built up a database and added hooks in the script view and download button to trigger a recording of the interaction. Figuring out what I would need to implement the recommender system and how to obtain/storestorage of the interaction information took me two days to complete.

Color Theme

I finally determined a color theme for the project. Avoiding brown, which worked poorly in my initial tests, I decided to go with my second choice of dark blue that also provides a sense of calmness and safety, exactly what I want users to feel.

Observations

With the implementation of a script add and view page it was much easier to rapidly develop the site. I had a much better understanding now of how to build React pages and how Laravel needed to be modified to efficiently handle the data. I used pre-built Bootstrap CSS to quickly create a proof of concept. While I don't plan on using a lot of the core bootstrap tools in the end it really helps with quick development. The design did not do wonders, but it was good enough.



-1

MonkeyScripts

Tutorial About

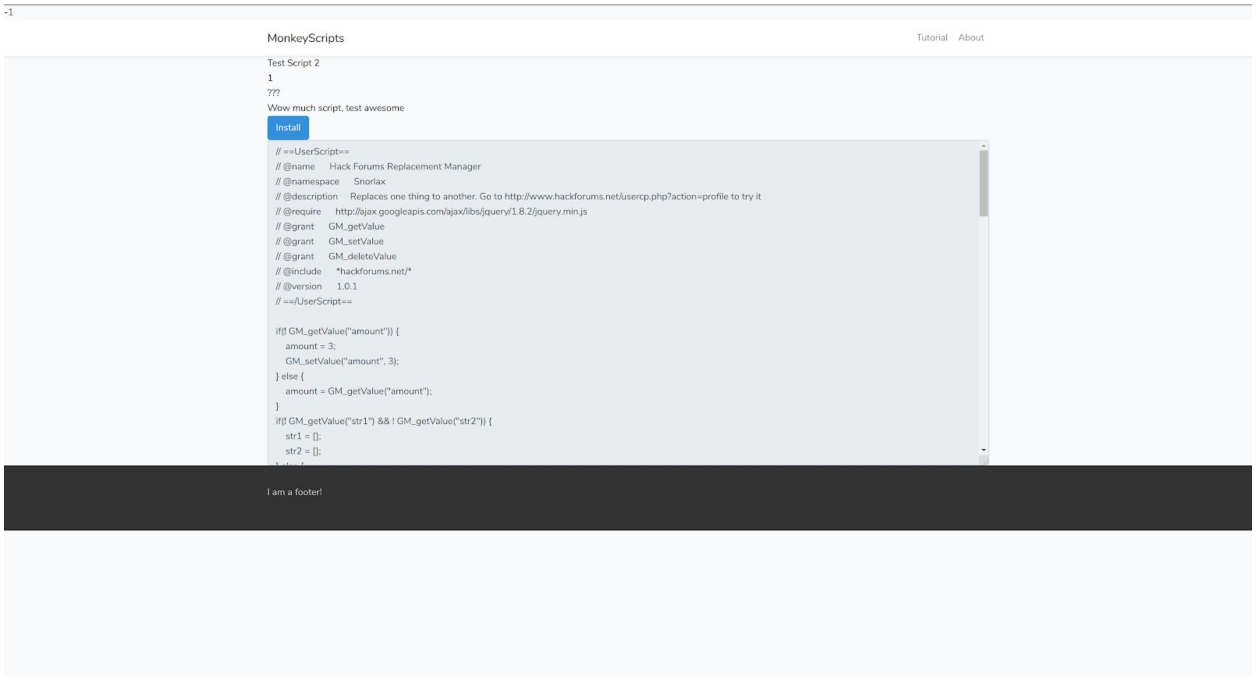
What are user scripts? How can they help?

User scripts are small browser extensions that can add features and fix bugs on your favorite sites!

Top Download	Top Rated
daDA - DDDDD	**daDA** - DDDDD
Test Scrip - Wow much script, test awesome	**Test Scrip** - Wow much script, test awesome
Test Script 2 - Wow much script, test awesome	**Test Script 2** - Wow much script, test awesome
Test Script 2 - Wow much script, test awesome	**Test Script 2** - Wow much script, test awesome
Test Script 3 - Wow much script, test awesome	**Test Script 3** - Wow much script, test awesome
Test Script 5 - Monkeyscripts is awesome!	**Test Script 5** - Monkeyscripts is awesome!
Test Script 6 - test test test	**Test Script 6** - test test test
Test Script 7 - so many tests, so little time!	**Test Script 7** - so many tests, so little time!
Test Script 8 - maybe this time redirects will work	**Test Script 8** - maybe this time redirects will work

Recently Updated

Editor's Choice



Test Script 2
1
???
Wow much script, test awesome

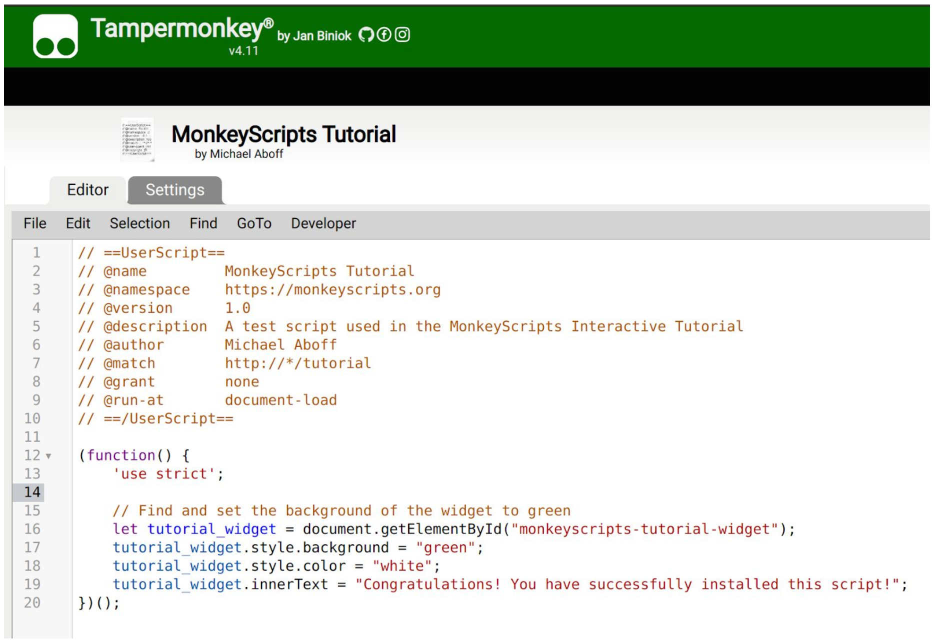
```
// ==UserScript==
// @name      Hack Forums Replacement Manager
// @namespace      Snorlax
// @description     Replaces one thing to another. Go to http://www.hackforums.net/usercp.php?action=profile to try it
// @require      http://ajax.googleapis.com/ajax/libs/jquery/1.8.2/jquery.min.js
// @grant     GM_getValue
// @grant     GM_setValue
// @grant     GM_deleteValue
// @include      *hackforums.net/*
// @version      1.0.1
// ==/UserScript==

if(! GM_getValue("amount")) {
    amount = 3;
    GM_setValue("amount", 3);
} else {
    amount = GM_getValue("amount");
}
if(! GM_getValue("str1") && ! GM_getValue("str2")) {
    str1 = [];
    str2 = [];
```

I am a footer!

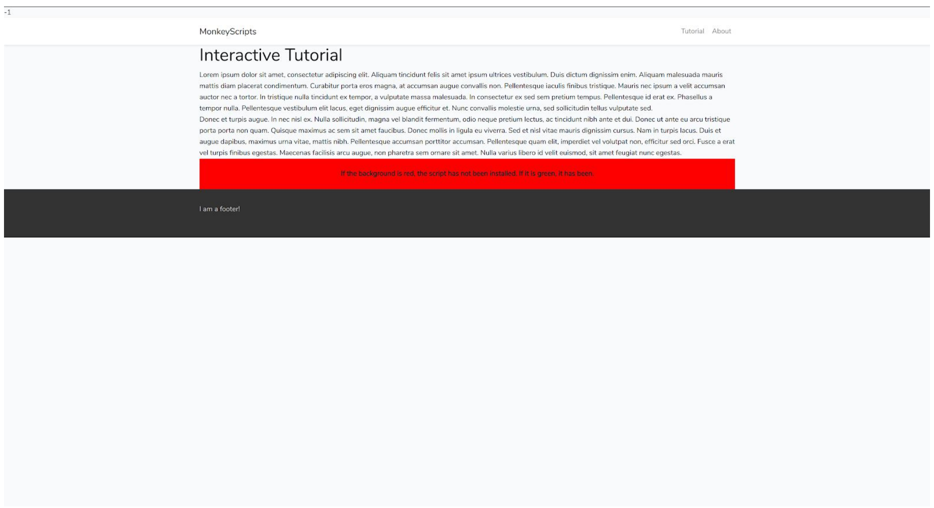
The Install button was actually surprisingly tricky. In order to install a user script, the user script manager has to detect that you are opening a raw .user.js file and it will prompt you to install it. My first though was that I could simply overwrite all of the visible components with the user script code and change the URL to .user.js. This did not work and I couldn’t trick it to do so. The only solution I could find was to have Laravel itself render a page outside of the React front-end with only the code itself whenever a user tried to load monkeyscripts.org/script/3.user.js. This workaround worked but actually ended up preventing me from easily recording user interactions.

When designing the interaction system, I decided I needed to prevent potential abuse, or at least make it more difficult to accomplish. The only way I have to do so is by restricting it to one interaction per registered user. Unfortunately collaborative filters require a large data set to start becoming useful, and I am not immediately expecting a large user base by the time the scheduled project has completed. One way I can help mitigate this issue is by incorporating even logged out users with an anonymous, unique identifier (a hash based off of their IP address, perhaps). This will have to be something I need to think further about. The other option would be to create a hybrid recommender system that incorporates something like script categories. I think for the sake of this project I will stick with a pure item-based collaborative filter recommender system, but it will be something I have to consider as we move forward.

I began the interactive part of the tutorial by developing a simple page and the user script which would modify an element on the page. The script development itself was pretty quick and already satisfying to use.



```
// ==UserScript==
// @name         MonkeyScripts Tutorial
// @namespace    https://monkeyscripts.org
// @version      1.0
// @description  A test script used in the MonkeyScripts Interactive Tutorial
// @author       Michael Aboff
// @match        http://*/tutorial
// @grant        none
// @run-at       document-load
// ==/UserScript==

(function() {
    'use strict';

    // Find and set the background of the widget to green
    let tutorial_widget = document.getElementById("monkeyscripts-tutorial-widget");
    tutorial_widget.style.background = "green";
    tutorial_widget.style.color = "white";
    tutorial_widget.innerText = "Congratulations! You have successfully installed this script!";
})();
```



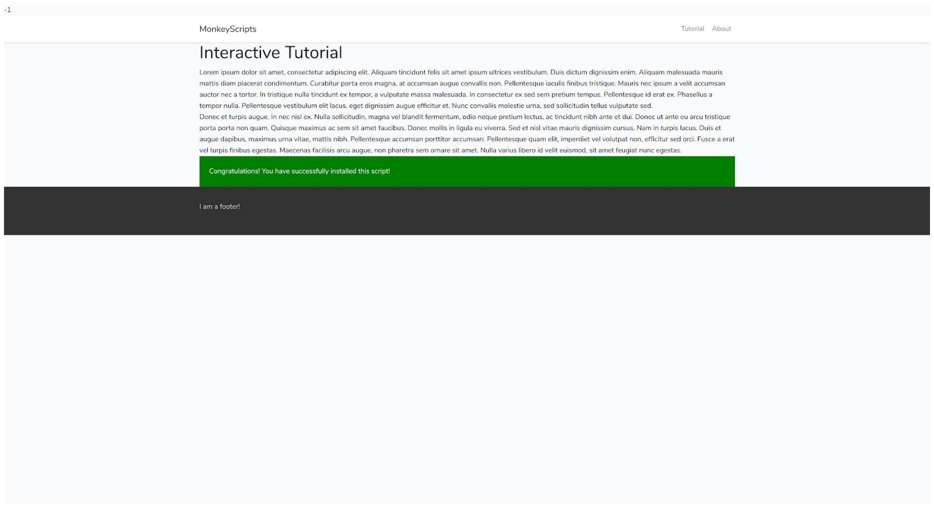
Interactive Tutorial

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam tincidunt felis sit amet ipsum ultrices vestibulum. Duis dictum dignissim enim. Aliquam malesuada mauris mattis diam placerat condimentum. Curabitur porta eros magna, at accumsan augue convallis non. Pellentesque iaculis finibus tristique. Mauris nec ipsum a velit accumsan auctor nec a tortor. In tristique nulla tincidunt ex tempor, a vulputate massa malesuada. In consectetur ex sed sem pretium tempus. Pellentesque id erat ex. Phasellus a tempor nulla. Pellentesque vestibulum elit lacus, eget dignissim augue efficitur et. Nunc convallis molestie urna, sed sollicitudin tellus vulputate sed.

Donec et turpis augue. In nec nisl ex. Nulla sollicitudin, magna vel blandit fermentum, odio neque pretium lectus, ac tincidunt nibh ante et dui. Donec ut ante eu arcu tristique porta porta non quam. Quisque maximus ac sem sit amet faucibus. Donec mollis in ligula eu viverra. Sed et nisl vitae mauris dignissim cursus. Nam in turpis lacus. Duis et augue dapibus, maximus urna vitae, mattis nibh. Pellentesque accumsan porttitor accumsan. Pellentesque quam elit, imperdiet vel volutpat non, efficitur sed orci. Fusce a erat vel turpis finibus egestas. Maecenas facilisis arcu augue, non pharetra sem ornare sit amet. Nulla varius libero id velit euismod, sit amet feugiat nunc egestas.

If the background is red, the script has not been installed. If it is green, it has been.

I am a footer!



Interactive Tutorial

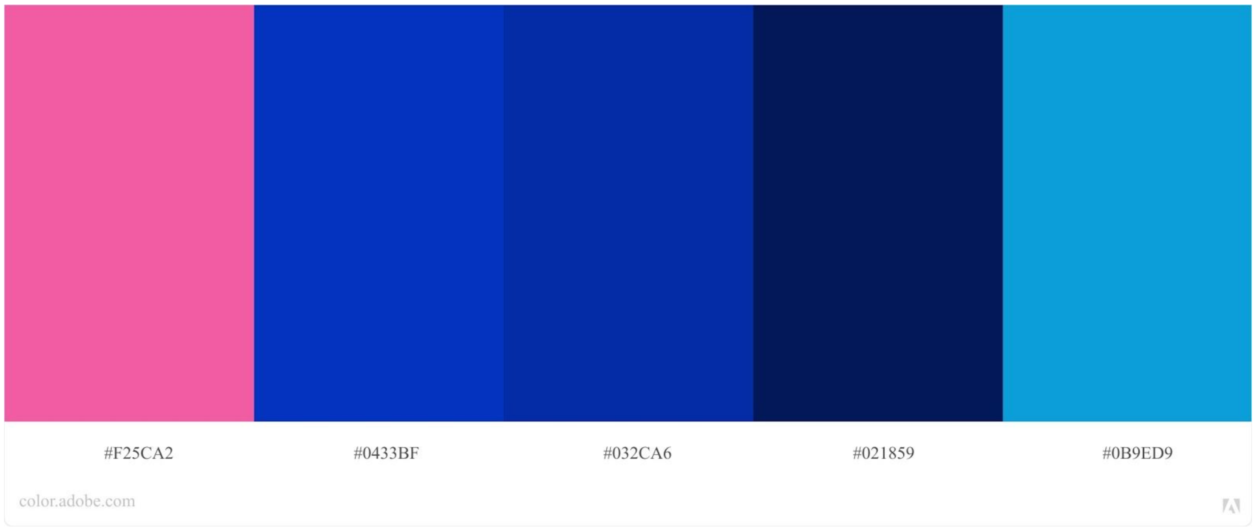
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam tincidunt felis sit amet ipsum ultrices vestibulum. Duis dictum dignissim enim. Aliquam malesuada mauris mattis diam placerat condimentum. Curabitur porta eros magna, at accumsan augue convallis non. Pellentesque iaculis finibus tristique. Mauris nec ipsum a velit accumsan auctor nec a tortor. In tristique nulla tincidunt ex tempor, a vulputate massa malesuada. In consectetur ex sed sem pretium tempus. Pellentesque id erat ex. Phasellus a tempor nulla. Pellentesque vestibulum elit lacus, eget dignissim augue efficitur et. Nunc convallis molestie urna, sed sollicitudin tellus vulputate sed.
Donec et turpis augue. In nec nisl ex. Nulla sollicitudin, magna vel blandit fermentum, odio neque pretium lectus, ac tincidunt nibh ante et dui. Donec ut ante eu arcu tristique porta porta non quam. Quisque maximus ac sem sit amet faucibus. Donec mollis in ligula eu viverra. Sed et nisl vitae mauris dignissim cursus. Nam in turpis lacus. Duis et augue dapibus, maximus urna vitae, mattis nibh. Pellentesque accumsan porttitor accumsan. Pellentesque quam elit, imperdiet vel volutpat non, efficitur sed orci. Fusce a erat vel turpis finibus egestas. Maecenas facilisis arcu augue, non pharetra sem ornare sit amet. Nulla varius libero id velit euismod, sit amet feugiat nunc egestas.

Congratulations! You have successfully installed this script!

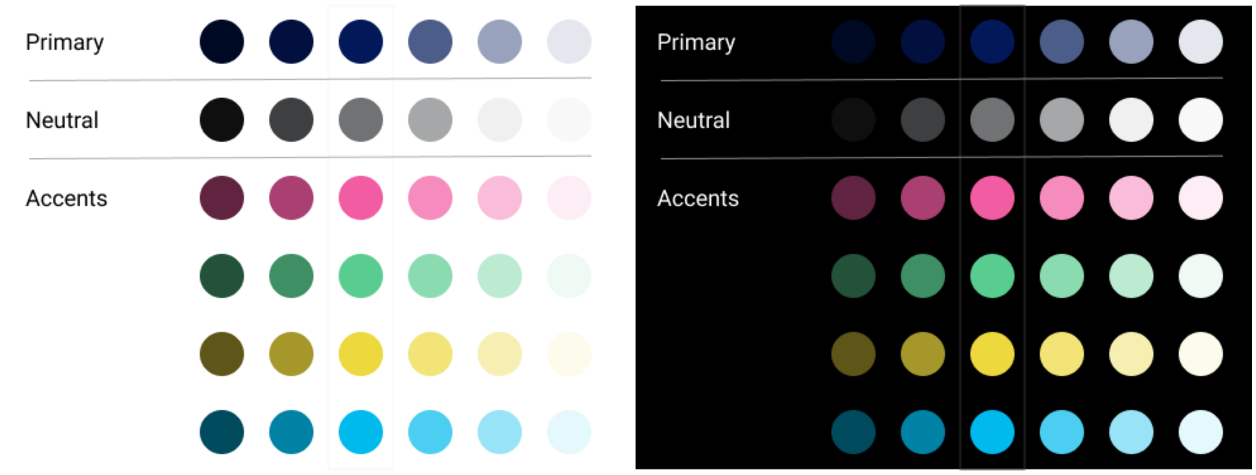
I am a footer!

This project I have been working hard to break out of a mental block I have where I feel the need to write every element of the code by myself. A prime example of this approach is with the search. As search is not a one of the two fundamental innovations of my project (tutorial and recommender), I found a third party service, Algolia, that will do searches for free. It has a convenient pre-built driver to work with Laravel's Scout plugin. Implementing it was easy to do and I surprised myself by how happy I ended up being by not having to develop my own search from scratch.

The color theme took quite some time to decide upon. I eventually went with a blue color with a purple splash. I feel like it has potential of working well with the final design, but only time will tell.



I then chose three of these colors and spun off a full palette that seems to work well with the design I have been building so far:



Reflection

I was very pleased with the progress I made this past iteration, making significant progress on many of the core portions of a standard website. However it is definitely not enough. My original plan expected me to have a fully working website by the end of the third iteration and there is still a lot more work to accomplish. Moving forward, there are a few must-have features that need to be implemented before I can release the site to the public and start getting their feedback:

- **Front-end and Back-end validation.** This is to make sure we only process the correct and full information.
- **Design Prototype.**
- **Design Implementation.**
- **Develop user scripts to populate the site with**
- **Recommender system.**
- **Tutorial.**
- **Google Analytics.**

I anticipate these stages to possibly take more time than I have left for this project. I will have to make significant progress over the next two weeks to complete in time.

Iteration 4

Plan

Iteration four is the final of our project's phases, covering weeks seven and eight. Due to the process taking longer than expected, I am starting these final weeks behind and need to make significant progress. During the last iteration's reflection, I defined 7 different must-have features that need to be completed before I start inviting users to join the website:

- **Front-end and Back-end validation.**
- **Design Prototype.**
- **Design Implementation.**
- **Develop user scripts to populate the site with**
- **Recommender system.**
- **Tutorial.**
- **Google Analytics.**

For the validation, we need to make sure that there is immediate user feedback when the user enters the wrong type of information. This will require reading form input and comparing it to expected values. Backend validation will also be required to prevent any malicious actors who might try to navigate around the first line of defense.

A design prototype is essential as building a design on the fly for a website of this complexity is nearly impossible. A cohesive design will be essential for a user friendly design. A bad design will turn away any potential users before they even reach the recommender or tutorial systems.

An empty site will also turn away potential users. As I have time, I need to create content that shows the user what possibilities there are to user scripts. This will take some time, and not as vital that I have a large repository yet.

The recommender system is what I am most concerned about. I have yet to implement this or know exactly how to do so, and I expect this to take the most amount of time to fully understand. Preliminary research suggested that summations will be involved. I will have to do more research and determine exactly which algorithm I'll want to implement.

Google analytics will be important, but not necessarily as vital as other components. The tool will be able to provide anonymized statistics about how many users join and how long they stay on the site for average.

Actions

Recommender

I started the first week, and spent the first 5 days of it researching and learning how to develop an item based collaborative filter. Using the previously created interactions table, I started off by building a pearson correlation summation to determine similarity between the different scripts. The formula would compare how the same users would interact with different scripts, determine a score based on how similar those interactions are. It becomes more accurate with more users and interactions.

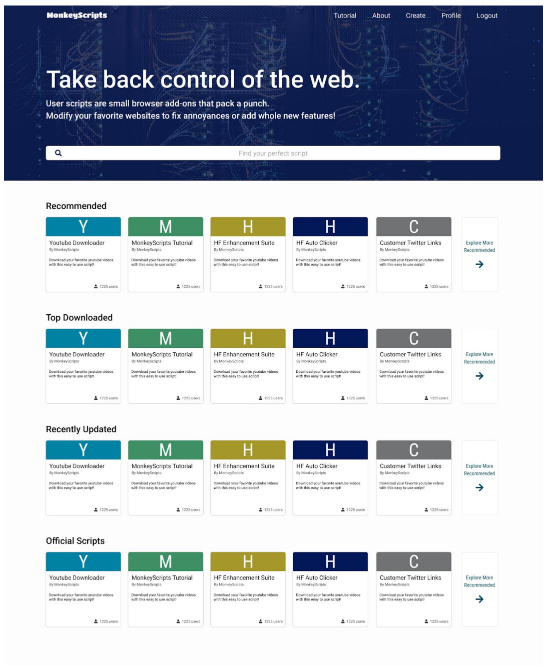
After determining how to calculate this score, I implemented a script that runs every minute, updating the scores so that users can get close to real time updates on what scripts are most related to other scripts. I would then multiply the pearson score against the total determined rating and use this to determine related scripts which would eventually be displayed on several different pages on the site.

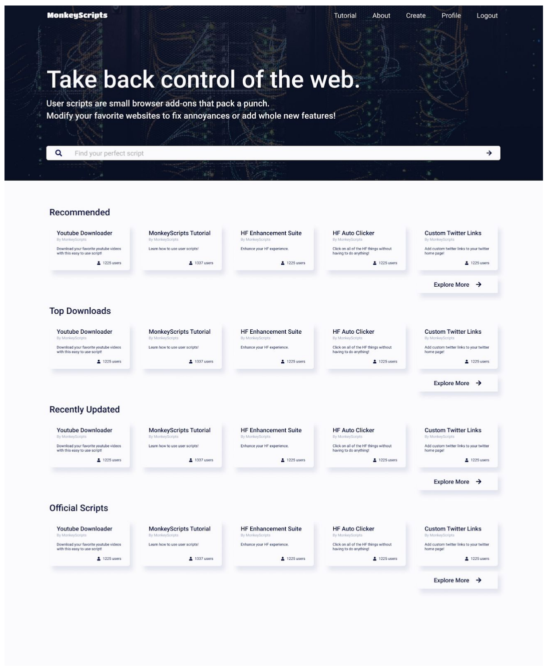
Anonymous Users

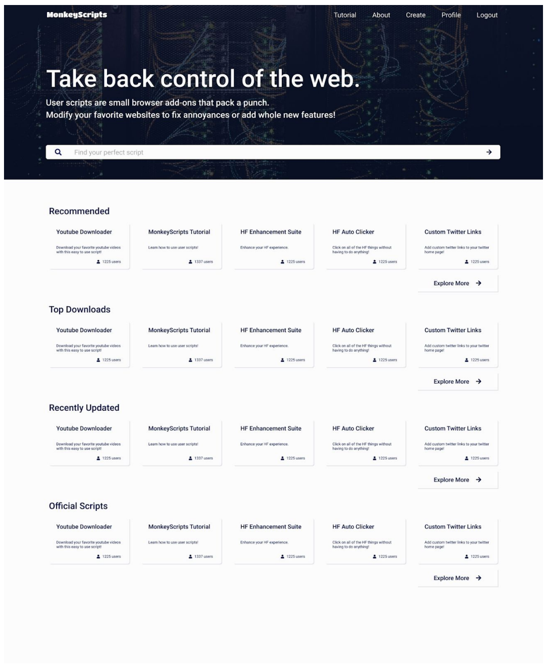
During the two weeks of this iteration, I realized I simply was not going to get enough information to have useful recommendations. To mitigate this, I created an anonymous user system. I combined the IP address and browser user agent together and hashed the result. I determined this would be unique enough for our purposes. This anonymous ID would be used to record whether or not the user viewed a script and/or pressed the download button.

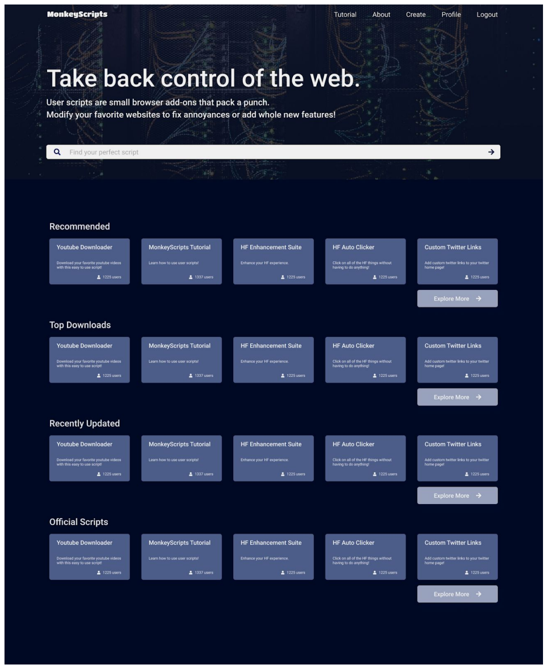
Design Prototyping

I was able to design the entire website. There was quite a revamp needed, especially how the script lists are displayed. This process took 3 days to create and several iterations:

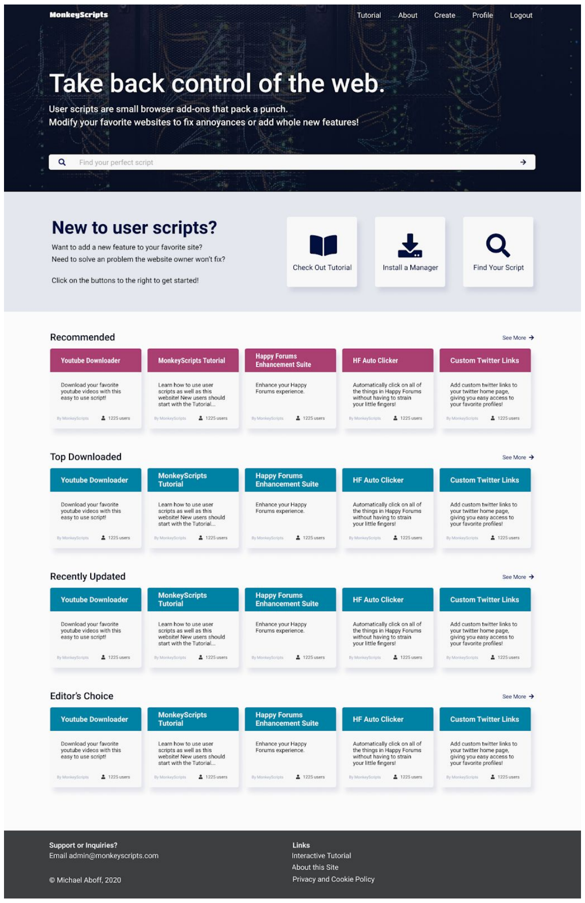
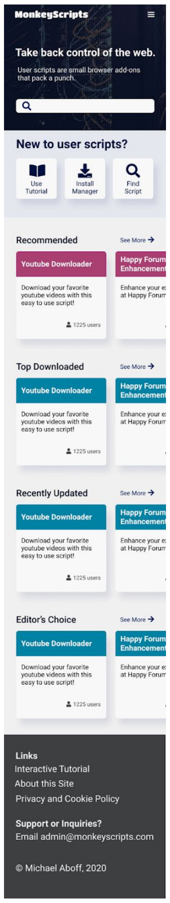

MonkeyScripts
Tutorial
About
Create
Profile
Logout
Take back control of the web.
User scripts are small browser add-ons that pack a punch.
Modify your favorite websites to fix annoyances or add whole new features!
Find your perfect script
Recommended
Y
Youtube Downloader
By MonkeyScripts
Download your favorite youtube videos with this easy to use script!
1225 users
M
MonkeyScripts Tutorial
By MonkeyScripts
Download your favorite youtube videos with this easy to use script!
1225 users
H
HF Enhancement Suite
By MonkeyScripts
Download your favorite youtube videos with this easy to use script!
1225 users
H
HF Auto Clicker
By MonkeyScripts
Download your favorite youtube videos with this easy to use script!
1225 users
C
Customer Twitter Links
By MonkeyScripts
Download your favorite youtube videos with this easy to use script!
1225 users
Explore More Recommended
Top Downloaded
Y
Youtube Downloader
M
MonkeyScripts Tutorial
H
HF Enhancement Suite
H
HF Auto Clicker
C
Customer Twitter Links
Explore More Recommended
Recently Updated
Y
Youtube Downloader
M
MonkeyScripts Tutorial
H
HF Enhancement Suite
H
HF Auto Clicker
C
Customer Twitter Links
Explore More Recommended
Official Scripts
Y
Youtube Downloader
M
MonkeyScripts Tutorial
H
HF Enhancement Suite
H
HF Auto Clicker
C
Customer Twitter Links
Explore More Recommended


MonkeyScripts
Tutorial
About
Create
Profile
Logout
Take back control of the web.
User scripts are small browser add-ons that pack a punch.
Modify your favorite websites to fix annoyances or add whole new features!
Find your perfect script
Recommended
Youtube Downloader
By MonkeyScripts
Download your favorite youtube videos with this easy to use script!
1225 users
MonkeyScripts Tutorial
By MonkeyScripts
Learn how to use user scripts!
1337 users
HF Enhancement Suite
By MonkeyScripts
Enhance your HF experience.
1225 users
HF Auto Clicker
By MonkeyScripts
Click on all of the HF things without having to do anything!
1225 users
Custom Twitter Links
By MonkeyScripts
Add custom twitter links to your twitter home page!
1225 users
Explore More
Top Downloads
Youtube Downloader
MonkeyScripts Tutorial
HF Enhancement Suite
HF Auto Clicker
Custom Twitter Links
Explore More
Recently Updated
Youtube Downloader
MonkeyScripts Tutorial
HF Enhancement Suite
HF Auto Clicker
Custom Twitter Links
Explore More
Official Scripts
Youtube Downloader
MonkeyScripts Tutorial
HF Enhancement Suite
HF Auto Clicker
Custom Twitter Links
Explore More


MonkeyScripts
Tutorial
About
Create
Profile
Logout
Take back control of the web.
User scripts are small browser add-ons that pack a punch.
Modify your favorite websites to fix annoyances or add whole new features!
Find your perfect script
Recommended
Youtube Downloader
Download your favorite youtube videos with this easy to use script!
1225 users
MonkeyScripts Tutorial
Learn how to use user scripts!
1337 users
HF Enhancement Suite
Enhance your HF experience.
1225 users
HF Auto Clicker
Click on all of the HF things without having to do anything!
1225 users
Custom Twitter Links
Add custom twitter links to your twitter home page!
1225 users
Explore More
Top Downloads
Youtube Downloader
MonkeyScripts Tutorial
HF Enhancement Suite
HF Auto Clicker
Custom Twitter Links
Explore More
Recently Updated
Youtube Downloader
MonkeyScripts Tutorial
HF Enhancement Suite
HF Auto Clicker
Custom Twitter Links
Explore More
Official Scripts
Youtube Downloader
MonkeyScripts Tutorial
HF Enhancement Suite
HF Auto Clicker
Custom Twitter Links
Explore More


MonkeyScripts
Tutorial
About
Create
Profile
Logout
Take back control of the web.
User scripts are small browser add-ons that pack a punch.
Modify your favorite websites to fix annoyances or add whole new features!
Find your perfect script
Recommended
Youtube Downloader
By MonkeyScripts
Download your favorite youtube videos with this easy to use script!
1225 users
MonkeyScripts Tutorial
By MonkeyScripts
Learn how to use user scripts!
1337 users
HF Enhancement Suite
By MonkeyScripts
Enhance your HF experience.
1225 users
HF Auto Clicker
By MonkeyScripts
Click on all of the HF things without having to do anything!
1225 users
Custom Twitter Links
By MonkeyScripts
Add custom twitter links to your twitter home page!
1225 users
Explore More
Top Downloads
Youtube Downloader
MonkeyScripts Tutorial
HF Enhancement Suite
HF Auto Clicker
Custom Twitter Links
Explore More
Recently Updated
Youtube Downloader
MonkeyScripts Tutorial
HF Enhancement Suite
HF Auto Clicker
Custom Twitter Links
Explore More
Official Scripts
Youtube Downloader
MonkeyScripts Tutorial
HF Enhancement Suite
HF Auto Clicker
Custom Twitter Links
Explore More

I finally decided on the below design after receiving some feedback from friends and family. Once this was decided, the rest of the design was much easier to develop:


MonkeyScripts
Tutorial
About
Create
Profile
Logout
Take back control of the web.
User scripts are small browser add-ons that pack a punch.
Modify your favorite websites to fix annoyances or add whole new features!
Find your perfect script
New to user scripts?
Want to add a new feature to your favorite site?
Need to solve an problem the website owner won't fix?
Click on the buttons to the right to get started!
Check Out Tutorial
Install a Manager
Find Your Script
Recommended
Top Downloaded
Recently Updated
Editor's Choice
See More
Youtube Downloader
MonkeyScripts Tutorial
Happy Forums Enhancement Suite
HF Auto Clicker
Custom Twitter Links
Support or Inquiries?
Email admin@monkeyscripts.com
© Michael Aboff, 2020
Links
Interactive Tutorial
About this Site
Privacy and Cookie Policy


Even though user scripts do not work well on mobile, I felt it was important that I design and develop a mobile version of the site as well, as many users may first see the site on their phones before deciding to use their computers.

Design Implementation

While starting to implement the design, I quickly realized that a lot of the naive decisions I made with React was not going to be sufficient for how I wanted to display the site. The majority of the front-end was redesigned with the new layout design in mind. Happily the process went quicker than I anticipated and took only 3 days to complete.

Tutorial System

I spent a few hours writing and testing the tutorial. I boiled the process down to two simple steps and tried to keep jargon as limited as possible. The work done previously with the script generation made this easy.

Script Development

For two days, I visited different user script forums looking for requests. In the end, I developed 6 scripts to be added to the site.

Validation

For frontend validation, one of the key features is to let a user know how many characters they have already typed into a form (to make sure they understand minimum and maximum character limits. The backend responses for form submissions were also better coordinated to provide similar types of response.

Google Analytics

It took one day to implement Google Analytics. I set up several custom events to be recorded on the site as well, including when a user visits the tutorial page specifically. I found that it did not record page visits very well due to the nature of my SPA implementation.

Deployment and Advertisement

I finally was able to deploy the website to a public server, configure nginx, and fix several initial bugs. I started sharing the site subtly providing links to user scripts for people who requested new ones to be created. Many users provided me unsolicited feedback, all of which was positive.

Observations

One of the core parts of my research is the hypothesis that content discovery is essential for user acquisition and retention. Developing a recommendation system is an essential component to this.

To seed the recommendation system with data, I need to record every interaction made by users on the site. To do this, I added a new database model called "Interactions". The first time a logged-in user visits a script page or requests a download of the script, it will be recorded as a boolean flag in the database.

Generally recommendation systems will base their recommendations off of the ratings given to an item. I do not have a rating system in place yet, so I decided to convert “viewed” and “downloaded” interactions into separate point values that could be added together.

As I learned during the literature review, there are two main categories types of collaborative filter recommendation systems: model based and memory based. Model based generally involves the use of machine learning, which is outside of the scope of this project (and honestly I definitely wouldn’t have time to learn how to use machine learning tools). Memory based would have to be the solution for me.

Pure memory based collaborative filtering then can be broken down into two different types: User-based and Item-based. User-based tries to determine which users are most similar to each other. If User A likes many of the same things that User B rates highly, then it can be assumed that they share similar tastes. Therefore if there is a new product that User B likes, the system can recommend the same product to User A. This is a great system and definitely preferred in many cases by large companies. The unfortunate part is that it requires a lot of user interaction, which is not something I can anticipate will happen quickly enough for this project.

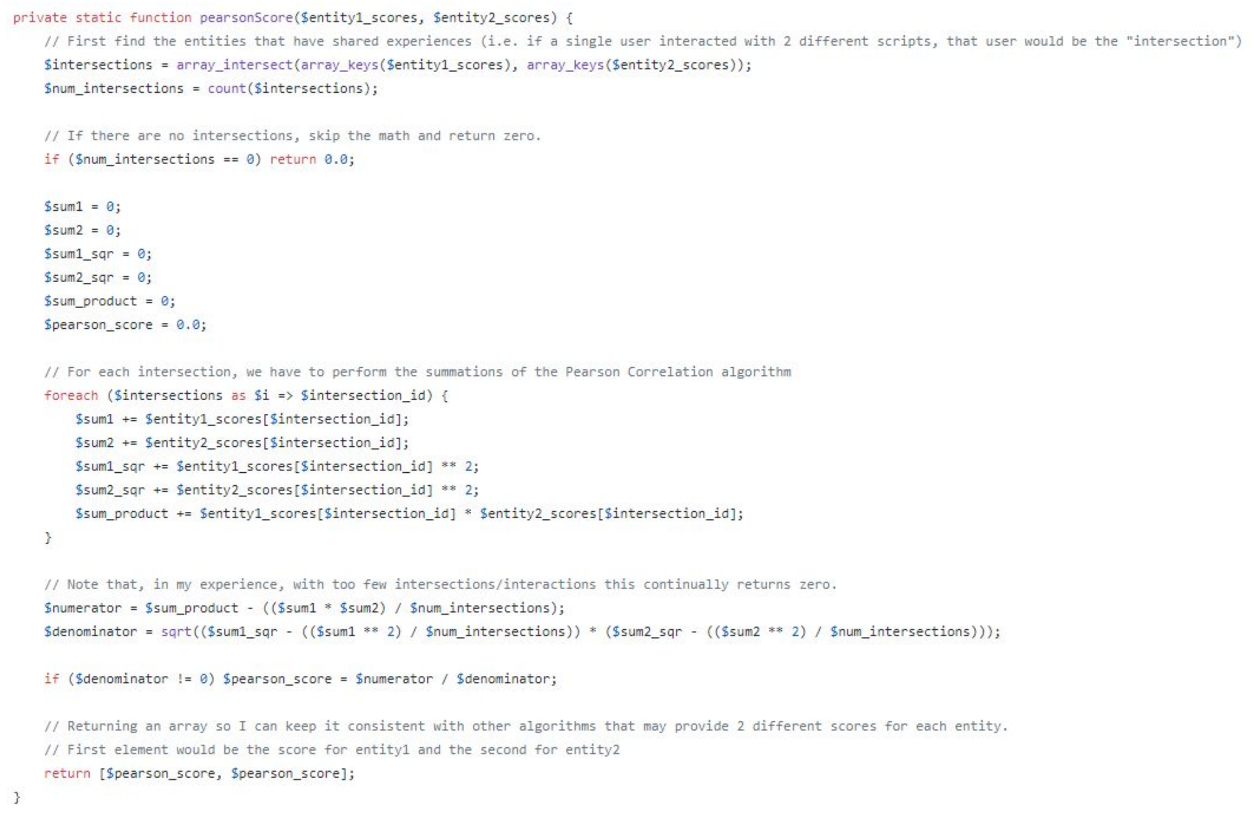
The alternative memory-based collaborative filtering is Item-based which attempts to find similar scripts based off of user interaction. If many users seem to interact with the same scripts, it would make sense to recommend these similar scripts to users. This is similar to how Amazon used to suggest items that other visitors would view. I determined that this would be able to handle the data sparsity problem better in my case than User based interaction.

To generate a similarity score, I set up a script to run every minute that will check every interaction and determine similarity scores for each. To generate these scores there will be two values collected:

1. The total interaction score for each script compared to every other script. The score would be equal to the total value of all viewed and downloaded interactions made by users on those scripts who also interacted with this script.
2. The second score calculated is the pearson correlation similarity score. This is done following the below formula which will give a value between -1 and 1 (1 being very similar).

$$\mathrm{simil}(x,y) = \frac{\sum\limits_{i \in I_{xy}} (r_{x,i} - \bar{r_x})(r_{y,i} - \bar{r_y})}{\sqrt{\sum\limits_{i \in I_{xy}} (r_{x,i} - \bar{r_x})^2} \sqrt{\sum\limits_{i \in I_{xy}} (r_{y,i} - \bar{r_y})^2}}$$

   What I discovered while implementing this is that you need several data points before it starts being useful. The formula will actually cancel itself until it has a significant number of similar scores to compare against.
3. I finally created a combined recommendation score by multiplying the score with the pearson correlation (after adding 1 to the correlation so I wouldn't just zero out all non similar recommendations, this is just a temporary solution until I get more users using the site).



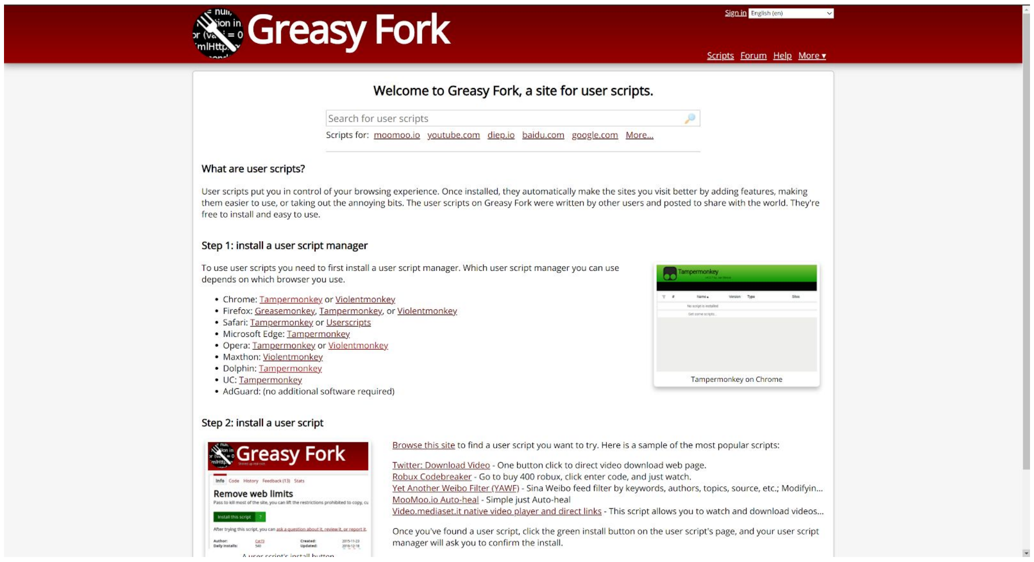
```
private static function pearsonScore($entity1_scores, $entity2_scores) {
    // First find the entities that have shared experiences (i.e. if a single user interacted with 2 different scripts, that user would be the "intersection")
    $intersections = array_intersect(array_keys($entity1_scores), array_keys($entity2_scores));
    $num_intersections = count($intersections);

    // If there are no intersections, skip the math and return zero.
    if ($num_intersections == 0) return 0.0;

    $sum1 = 0;
    $sum2 = 0;
    $sum1_sqr = 0;
    $sum2_sqr = 0;
    $sum_product = 0;
    $pearson_score = 0.0;

    // For each intersection, we have to perform the summations of the Pearson Correlation algorithm
    foreach ($intersections as $i => $intersection_id) {
        $sum1 += $entity1_scores[$intersection_id];
        $sum2 += $entity2_scores[$intersection_id];
        $sum1_sqr += $entity1_scores[$intersection_id] ** 2;
        $sum2_sqr += $entity2_scores[$intersection_id] ** 2;
        $sum_product += $entity1_scores[$intersection_id] * $entity2_scores[$intersection_id];
    }

    // Note that, in my experience, with too few intersections/interactions this continually returns zero.
    $numerator = $sum_product - (($sum1 * $sum2) / $num_intersections);
    $denominator = sqrt(($sum1_sqr - (($sum1 ** 2) / $num_intersections)) * ($sum2_sqr - (($sum2 ** 2) / $num_intersections)));

    if ($denominator != 0) $pearson_score = $numerator / $denominator;

    // Returning an array so I can keep it consistent with other algorithms that may provide 2 different scores for each entity.
    // First element would be the score for entity1 and the second for entity2
    return [$pearson_score, $pearson_score];
}
```

Determining similar scripts is only the first step, now I have to be able to determine which scripts to recommend to the user when they are browsing the site. Due to time (and expected sparse data) I came up with this algorithm:

1. Look at the user's X most recently interacted with scripts.
    a. If there are not enough information or the user is not logged in, simply start off with a generic list of the top 10 most downloaded scripts.
2. For each of the selected scripts, find their Y most similar scripts (2 by default)
3. Remove any duplicates from this list.
4. Randomize the list and display the top 5.

While this isn’t really following the actual steps of recommending scripts to users, I felt that this at least let us provide some usable information until the user base and user interactions was enough where we can provide a more robust recommendation system.

While developing the site design, it was essential that it was welcoming and easier to understand than the currently existing alternatives out there. The two major competitors both struggle with a lack of whitespace, typography, color schemes, and clear direction for new users:



Welcome to Greasy Fork, a site for user scripts.

Search for user scripts

Scripts for: moomoo.io youtube.com diep.io baidu.com google.com More...

What are user scripts?

User scripts put you in control of your browsing experience. Once installed, they automatically make the sites you visit better by adding features, making them easier to use, or taking out the annoying bits. The user scripts on Greasy Fork were written by other users and posted to share with the world. They're free to install and easy to use.

Step 1: install a user script manager

To use user scripts you need to first install a user script manager. Which user script manager you can use depends on which browser you use.

- Chrome: Tampermonkey or Violentmonkey
- Firefox: Greasemonkey, Tampermonkey, or Violentmonkey
- Safari: Tampermonkey or Userscripts
- Microsoft Edge: Tampermonkey
- Opera: Tampermonkey or Violentmonkey
- Maxthon: Violentmonkey
- Dolphin: Tampermonkey
- UC: Tampermonkey
- AdGuard: (no additional software required)

Tampermonkey on Chrome

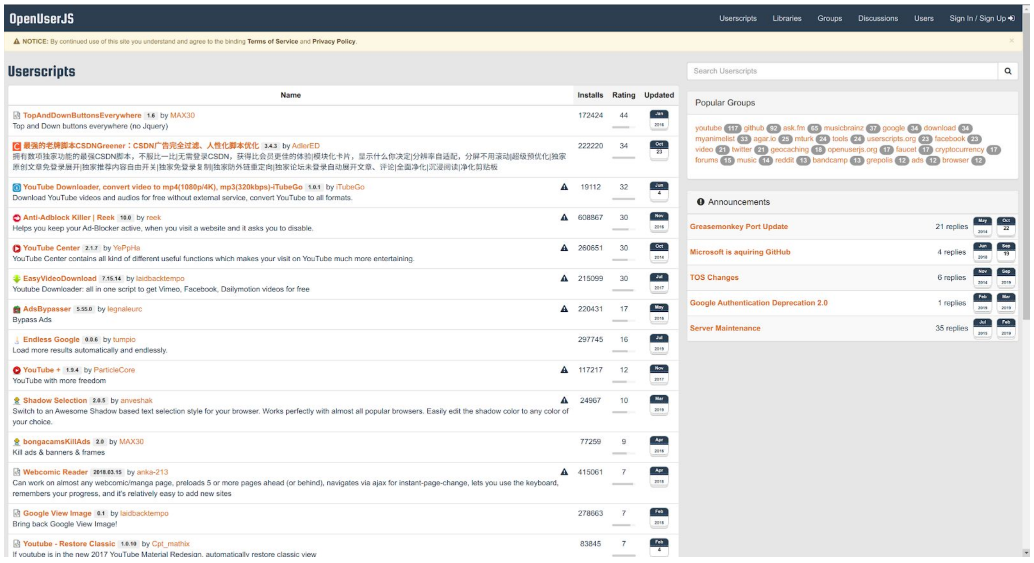
Step 2: install a user script

Remove web limits

Browse this site to find a user script you want to try. Here is a sample of the most popular scripts:

Twitter: Download Video - One button click to direct video download web page.
Robux Codebreaker - Go to buy 400 robux, click enter code, and just watch.
Yet Another Weibo Filter (YAWF) - Sina Weibo feed filter by keywords, authors, topics, source, etc.; Modifyin...
MooMoo.io Auto-heal - Simple just Auto-heal
Video.mediaset.it native video player and direct links - This script allows you to watch and download videos...

Once you've found a user script, click the green install button on the user script's page, and your user script manager will ask you to confirm the install.



NOTICE: By continued use of this site you understand and agree to the binding Terms of Service and Privacy Policy.

Userscripts

Name		Installs	Rating	Updated
TopAndDownButtonsEverywhere 1.6 by MAX30 Top and Down buttons everywhere (no Jquery)		172424	44	Jan 2016
最强的老牌脚本CSDNGreener：CSDN广告完全过滤、人性化脚本优化 3.4.3 by AdlerED 拥有数项独家功能的最强CSDN脚本，不服比一比|无需登录CSDN，获得比会员更佳的体验|模块化卡片，显示什么你决定|分辨率自适配，分屏不用滚动|超级预优化|独家原创文章免登录展开|独家推荐内容自由开关|独家免登录复制|独家防外链重定向|独家论坛未登录自动展开文章、评论|全面净化|沉浸阅读|净化剪贴板		222220	34	Oct 23
YouTube Downloader, convert video to mp4(1080p/4K), mp3(320kbps)-iTubeGo 1.0.1 by iTubeGo Download YouTube videos and audios for free without external service, convert YouTube to all formats.	⚠	19112	32	Jun 4
Anti-Adblock Killer | Reek 10.0 by reek Helps you keep your Ad-Blocker active, when you visit a website and it asks you to disable.	⚠	608867	30	Nov 2016
YouTube Center 2.1.7 by YePpHa YouTube Center contains all kind of different useful functions which makes your visit on YouTube much more entertaining.	⚠	260651	30	Oct 2014
EasyVideoDownload 7.15.14 by laidbacktempo Youtube Downloader: all in one script to get Vimeo, Facebook, Dailymotion videos for free	⚠	215099	30	Jul 2017
AdsBypasser 5.55.0 by legnaleurc Bypass Ads	⚠	220431	17	May 2016
Endless Google 0.0.6 by tumpio Load more results automatically and endlessly.		297745	16	Jul 2019
YouTube + 1.9.4 by ParticleCore YouTube with more freedom	⚠	117217	12	Nov 2017
Shadow Selection 2.0.5 by anveshak Switch to an Awesome Shadow based text selection style for your browser. Works perfectly with almost all popular browsers. Easily edit the shadow color to any color of your choice.	⚠	24967	10	Mar 2019
bongacamsKillAds 2.0 by MAX30 Kill ads & banners & frames		77259	9	Apr 2016
Webcomic Reader 2018.03.15 by anka-213 Can work on almost any webcomic/manga page, preloads 5 or more pages ahead (or behind), navigates via ajax for instant-page-change, lets you use the keyboard, remembers your progress, and it's relatively easy to add new sites	⚠	415061	7	Apr 2018
Google View Image 0.1 by laidbacktempo Bring back Google View Image!		278663	7	Feb 2018
Youtube - Restore Classic 1.0.10 by Cpt_mathix If youtube is in the new 2017 YouTube Material Redesign, automatically restore classic view		83845	7	Feb 4

Search Userscripts

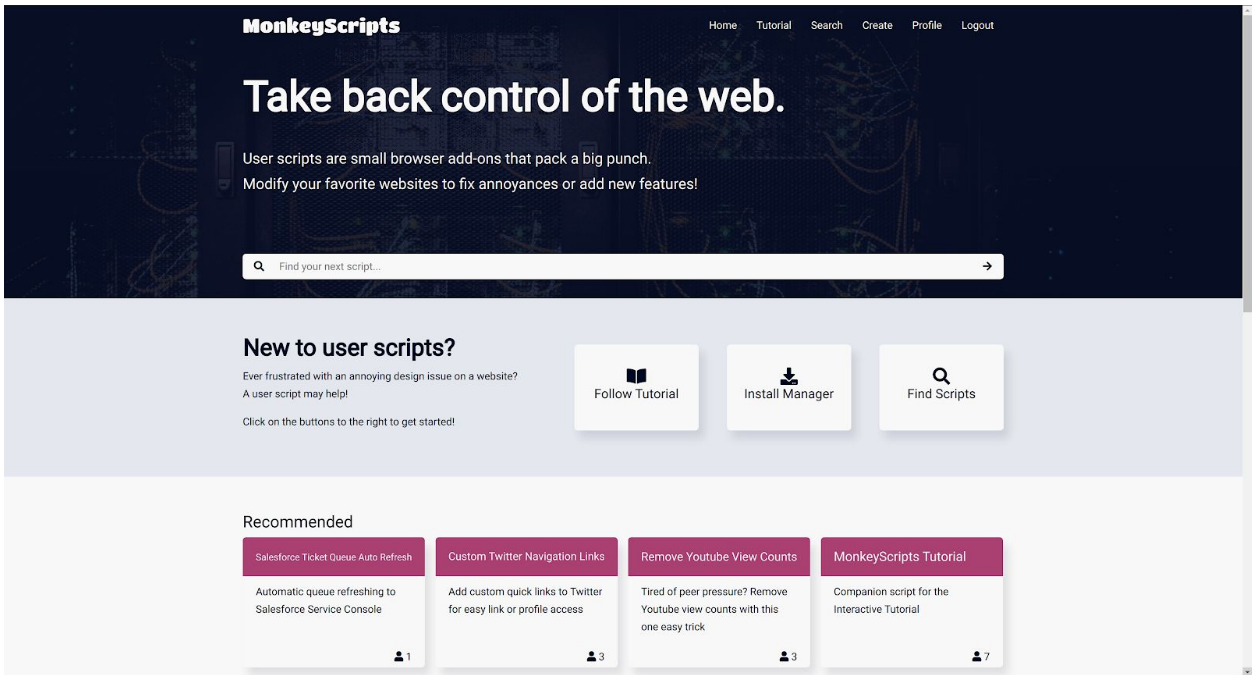
Popular Groups

youtube 117 github 92 ask.fm 65 musicbrainz 37 google 34 download 34 myanimelist 33 agar.io 25 mturk 24 tools 24 userscripts.org 23 facebook 23 video 21 twitter 21 geocaching 18 openuserjs.org 17 faucet 17 cryptocurrency 17 forums 15 music 14 reddit 13 bandcamp 13 grepolis 12 ads 12 browser 12

Announcements

Greasemonkey Port Update	21 replies	May 2014	Oct 22
Microsoft is aquiring GitHub	4 replies	Jun 2018	Sep 19
TOS Changes	6 replies	Nov 2014	Sep 2019
Google Authentication Deprecation 2.0	1 replies	Feb 2019	Mar 2019
Server Maintenance	35 replies	Jul 2015	Feb 2019

My design focused more on minimalism and providing the customer higher quality recommendations immediately while also encouraging users to learn how to use the product. With the below image of the final website homepage implementation that a large portion of the home page is devoted to explaining what user scripts are as well as how to use them:



Take back control of the web.

User scripts are small browser add-ons that pack a big punch.
Modify your favorite websites to fix annoyances or add new features!

New to user scripts?

Ever frustrated with an annoying design issue on a website?
A user script may help!

Click on the buttons to the right to get started!

Recommended

Salesforce Ticket Queue Auto Refresh
Automatic queue refreshing to Salesforce Service Console
1

Custom Twitter Navigation Links
Add custom quick links to Twitter for easy link or profile access
3

Remove Youtube View Counts
Tired of peer pressure? Remove Youtube view counts with this one easy trick
3

MonkeyScripts Tutorial
Companion script for the Interactive Tutorial
7

Please see Appendix D for more screenshots of the final design.

There were several features I was unable to implement in the end, that were high on my to-do list and will need to be addressed as I continue to build up this website and obtain more feedback on how people like to use it. Some of these important but not show stopping features include:

User email verification management. Currently users can sign in with literally any email they want, there is no verification happening or throttling. This is a huge potential issue and a large reason why I have not been able to heavily advertise the site to new users yet. Unfortunately I simply ran out of time while working on a fix for this.

Administration tools. As of right now there are no administrative tools to help ban or edit users. I determined this would be nice to have, but not a critical feature as I would be able to perform direct database edits as needed (and I am regularly capturing backups).

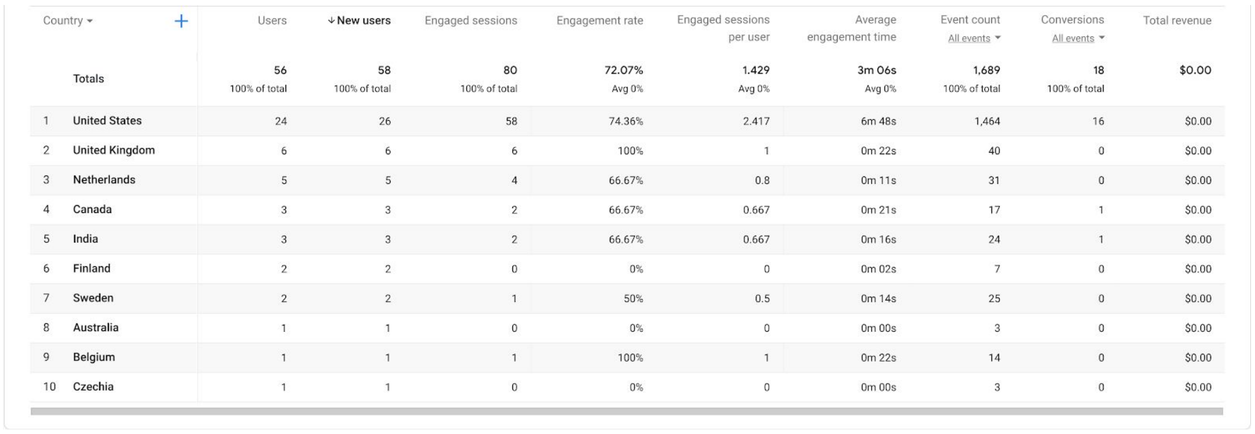
Reflection

This was a really difficult two weeks for me. Somehow I was able to complete the must-haves I listed, which should have taken closer to 20 days to complete based off of initial estimates. Unfortunately I was unable to have enough time to polish the site and really fill out the site with content that may have made it more approachable to users.

I had 58 new users visit the site over the final three days I had been lightly advertising. Several users reached out to me, providing mild constructive criticism regarding design choices (such as page hierarchy and lack of breadcrumb.

I felt that the tutorial was designed well. It includes a video instruction on how to install a 3rd party software, it includes written instructions, and finally includes an interactive component where a user has to navigate to a specific test script, install it, and come back to the tutorial page to confirm success. Several users informed me that this was a very good way to proceed and was clear.

While incredibly stressful, filled with very late nights, completing the website was definitely a satisfying and exciting experience. I look forward to being able to sleep more than 2 hours a night



	Country	Users	New users	Engaged sessions	Engagement rate	Engaged sessions per user	Average engagement time	Event count All events	Conversions All events	Total revenue
	Totals	56 100% of total	58 100% of total	80 100% of total	72.07% Avg 0%	1.429 Avg 0%	3m 06s Avg 0%	1,689 100% of total	18 100% of total	$0.00
1	United States	24	26	58	74.36%	2.417	6m 48s	1,464	16	$0.00
2	United Kingdom	6	6	6	100%	1	0m 22s	40	0	$0.00
3	Netherlands	5	5	4	66.67%	0.8	0m 11s	31	0	$0.00
4	Canada	3	3	2	66.67%	0.667	0m 21s	17	1	$0.00
5	India	3	3	2	66.67%	0.667	0m 16s	24	1	$0.00
6	Finland	2	2	0	0%	0	0m 02s	7	0	$0.00
7	Sweden	2	2	1	50%	0.5	0m 14s	25	0	$0.00
8	Australia	1	1	0	0%	0	0m 00s	3	0	$0.00
9	Belgium	1	1	1	100%	1	0m 22s	14	0	$0.00
10	Czechia	1	1	0	0%	0	0m 00s	3	0	$0.00

Reflective Statement

My original vision for this project was to revitalize a stagnating community surrounding user scripts. As I continued through the actual iterations themselves and continued building the site, I realized that no website built in two months would be able to do that alone. It would require continued development over many more months, as well as community buy-in before it would be able to make a difference. Where this work is important, I believe, is as a feasibility study, a pilot that shows that users are receptive to a modern and friendly design. Clear, simple, and interactive explanations go over well with users who are unfamiliar with this.

The largest issue with this project was the choice early on to learn a new technology stack. While learning these tools is invaluable for myself and my future career, it only served to add significant delays to the schedule that I could not afford with all of the additional work that needed to be done. With more time and polish the product, the more confident I would be that this could be a very influential site in the community.

Throughout this project, I learned how recommendation systems prop up the largest companies in the world. I have learned how important it is to teach and encourage learners through a variety of methods and techniques to help them learn complicated topics. I believe that I was able to implement both of these lessons into the site and prove that there are users interested. Further research and study should be done to further refine the techniques used.

I learned a lot and while there were mistakes made along the way that I will learn from, I would definitely do it again and I am confident it will help me with my own future career.

References

Andriotis, N. (2018, November 28). Customer Education: The Benefits And Best Practices. Retrieved August 31, 2020, from https://www.talentlms.com/blog/customer-education-benefits-best-practices/

Cao, J. (2015, April 7). Web design color theory: How to create the right emotions with color in web design. Retrieved October 21, 2020, from https://thenextweb.com/dd/2015/04/07/how-to-create-the-right-emotions-with-color-in-web-design/

Chapman, C. (2010, January 28). Color Theory for Designers, Part 1: The Meaning of Color. Retrieved October 22, 2020, from https://www.smashingmagazine.com/2010/01/color-theory-for-designers-part-1-the-meaning-of-color/

Council, Y. (2017, December 06). 15 Methods for Educating Customers About Your New Services and Products. Retrieved August 31, 2020, from https://www.huffpost.com/entry/15-methods-for-educating_b_10968478?guccounter=1

Craig, W. (2015, April 10). Don't Market To Your Customers; Educate Them Instead. Retrieved August 31, 2020, from https://www.forbes.com/sites/williamcraig/2015/04/10/dont-market-to-your-customers-educate-them-instead/

Bixby, D. W. (2018). Culture and Proficiency: Training for Proficiency in a Global Environment. *Product training for the technical expert : The art of developing and delivering hands-on learning*. (pp 161-166) ProQuest Ebook Central. Retrieved September 28, 2020, from https://ebookcentral.proquest.com/lib/wilmcoll-ebooks/detail.action?docID=5215305

Bobadilla, J., Ortega, F., Hernando, A., & Arroyo, Á. (2012). A balanced memory-based collaborative filtering similarity measure. *International Journal of Intelligent Systems*, 27(10), 939–946. Retrieved September 27, 2020, from https://doi-org.mylibrary.wilmu.edu/10.1002/int.21556

Brinkmann, M. (2014, July 23). Userscripts.org down for good? Here are alternatives - gHacks Tech News. Retrieved September 22, 2020, from https://www.ghacks.net/2014/05/09/userscripts-org-good-alternatives/

Carmel, Y., & Patt-Shamir, B. (2016). Comparison-based interactive collaborative filtering. *Theoretical Computer Science*, *628*, 40–49. Retrieved September 28, 2020, from https://doi.org/10.1016/j.tcs.2016.03.010

Collatto, D. C., Dresch, A., Lacerda, D. P., & Bentz, I. G. (2018). Is action design research indeed necessary? analysis and synergies between action research and design science research. Systemic Practice and Action Research, 31(3), 239–267. https://doi.org/10.1007/s11213-017-9424-9

Extensionmonitor.com. (2019, August 2). Breaking Down the Chrome Web Store: An exploratory analysis of extensions (part 1). Retrieved September 22, 2020, from https://extensionmonitor.com/blog/breaking-down-the-chrome-web-store-part-1

Gomez-Uribe, C. A., & Hunt, N. (2015). The Netflix Recommender System: Algorithms, Business Value, and Innovation. *ACM Transactions on Management Information Systems, 6*(4), 13:1-13:19. Retrieved September 27, 2020, from https://doi.org/10.1145/2843948.

Hardy, W., & Rodman, J. (2016). Action research. *Military Review*, *96*(1).

King, K. P. (2017). e-Learning Models: Distance, Mobile, Virtual, and Informal Learning. *Technology and innovation in adult learning*. (pp 203-219) ProQuest Ebook Central. Retrieved September 25, 2020, from https://ebookcentral.proquest.com/lib/wilmcoll-ebooks/reader.action?docID=4816176

McNiff, J. (2016). You and your action research project (Fourth). Routledge. Retrieved from https://wilmu.on.worldcat.org/oclc/925410866

Pilgrim, M. (2009). Greasemonkey Hacks: Tips & Tools for Remixing the Web with Firefox. United States: O'Reilly Media.

Pilgrim, M. (2005, September 9). Dive Into Greasemonkey. Retrieved September 22, 2020, from http://web.archive.org/web/20110726000633/http://diveintogreasemonkey.org/

Rao U.H., Nayak U. (2014) Key Concepts and Principles. In: The InfoSec Handbook. Apress, Berkeley, CA. https://doi-org.mylibrary.wilmu.edu/10.1007/978-1-4302-6383-8_3

Rutigliano, K. , & Video Education Australasia (Producers), & Clarry, E. (Director). (2010). Adult learning 1: Principles. [Video/DVD] Video Education Australasia. Retrieved September 24, 2020, from https://video-alexanderstreet-com.mylibrary.wilmu.edu/watch/adult-learning-1-principles

Rutigliano K., & Video Education Australasia (Producers), & Clarry, E. (Director). (2010). Adult learning 2: Styles. [Video/DVD] Video Education Australasia. Retrieved September 24, 2020, from https://video-alexanderstreet-com.mylibrary.wilmu.edu/watch/adult-learning-2-styles

Stowell, F., & Cooray, S. (2017). Virtual action research for virtual organisations? Systemic Practice and Action Research, 30(2), 117–143. https://doi.org/10.1007/s11213-016-9384-5

Ulvik, M., Riese, H., & Roness, D. (2018). Action research - connecting practice and theory. *Educational Action Research*, *26*(2), 273–287. Retrieved from https://doi.org/10.1080/09650792.2017.1323657

Wlodkowski, R. J., & Ginsberg, M. B. (2017). Characteristics and Skills of a Motivating Instructor: The Five Pillars. *Enhancing adult motivation to learn : A comprehensive guide for teaching all adults*. (pp 47-106). ProQuest Ebook Central. Retrieved September 25, 2020, from https://ebookcentral.proquest.com/lib/wilmcoll-ebooks/detail.action?docID=4983754.

Wong, C. (2020, June 17). How to Choose Good Website Color Schemes. Retrieved October 22, 2020, from https://www.websitebuilderexpert.com/designing-websites/how-to-choose-color-for-your-website/

Appendix A: Glossary

Browser: Web browsers are applications that allow users to request and interact with websites. Popular browsers include Google Chrome, Mozilla Firefox, Microsoft Edge, Apple's Safari, and Opera.

Google Trends: A tool provided by Google that provides relative comparisons of interest in provided topics. Results are based off of searches made with the Google Search engine. Results are presented on a scale of 0-100.

User Script: A cross browser add-on that modifies or augments a website.

User Script Manager: A browser specific add-on that executes the user script. Common managers include Greasemonkey, Tampermonkey, and Violentmonkey.

Appendix B: Google Trends

Figure B.1 - Google Trends report for the terms "userscript", "userscripts", "user script", "userscripts".

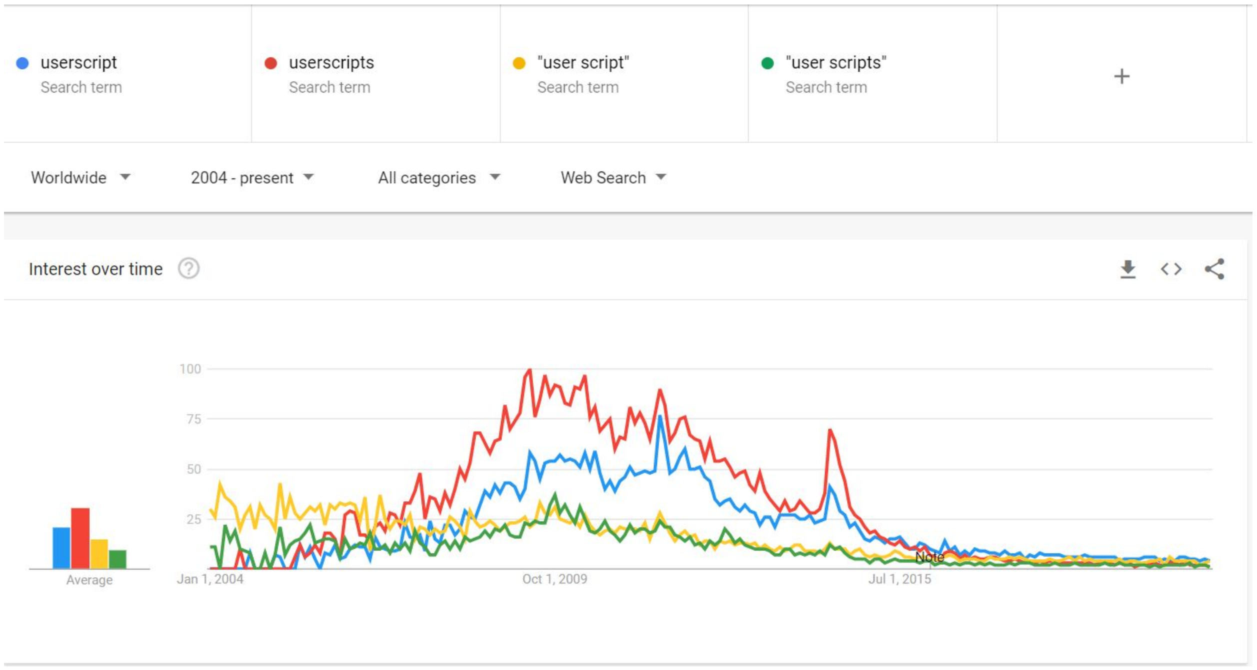


Figure B.2 - Google Trends report for the user script managers "greasemonkey", "tampermonkey", and "violentmonkey".

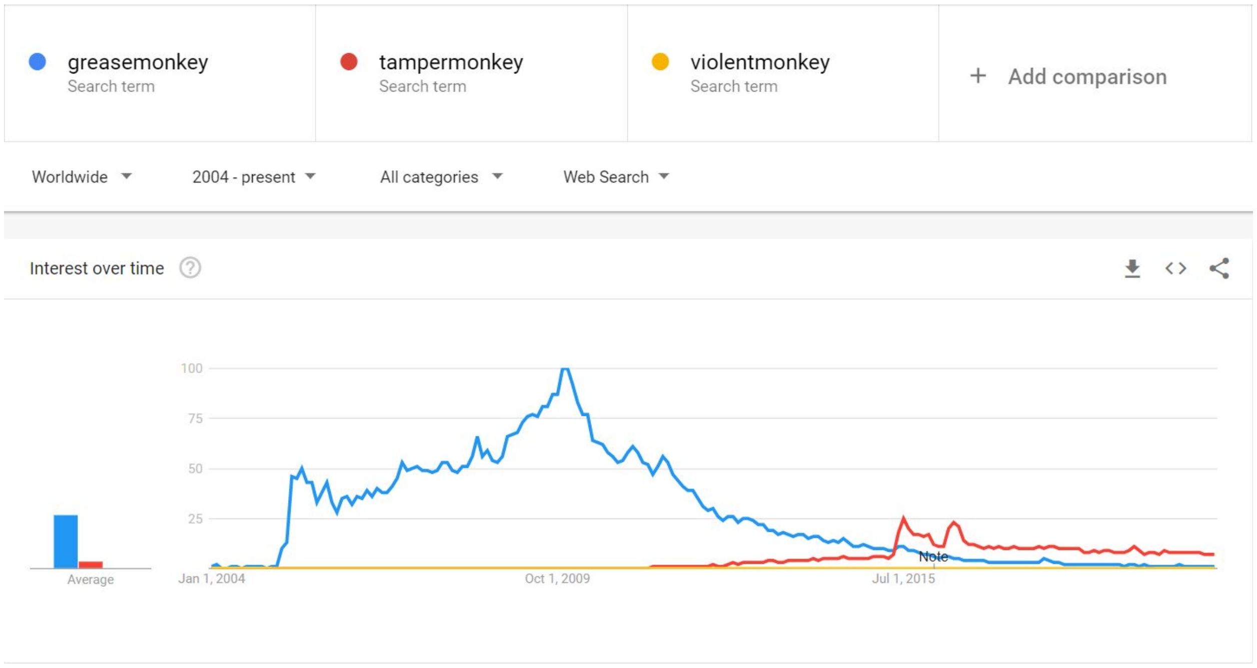


Figure B.3 - Google Trends report for current popular repositories "openuserjs" and "greasyfork"

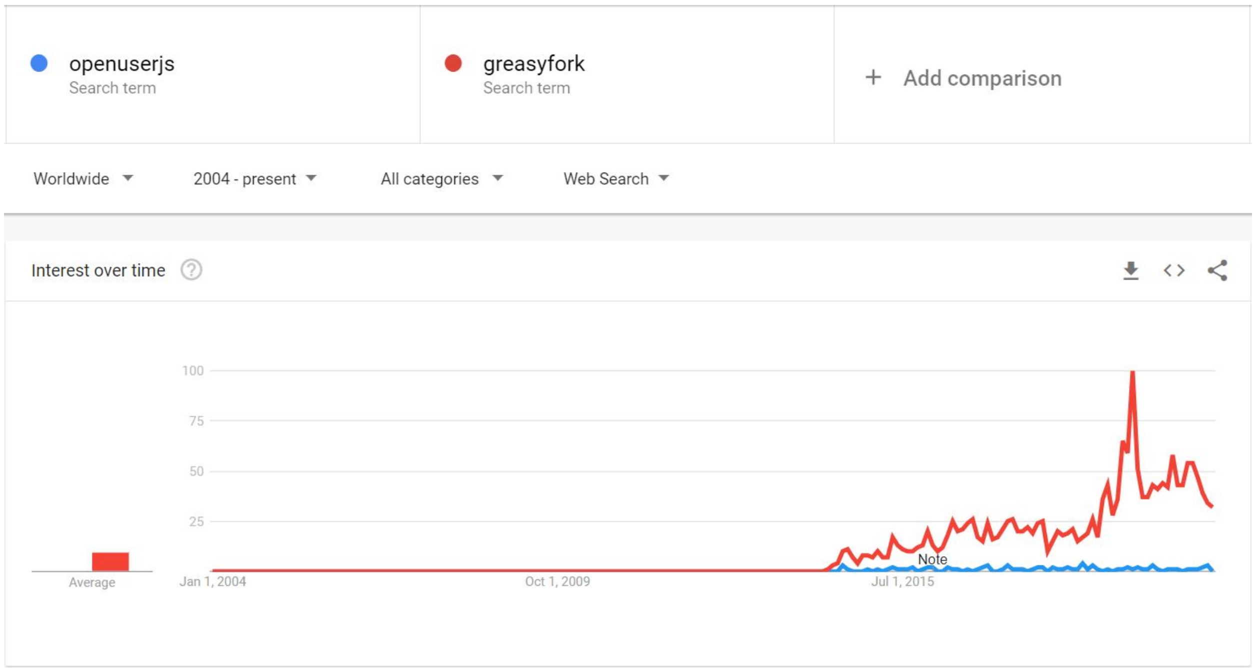


Figure B.4 - Google Trends report for current popular repository URLs “openuserjs.org” and “greasyfork.org”

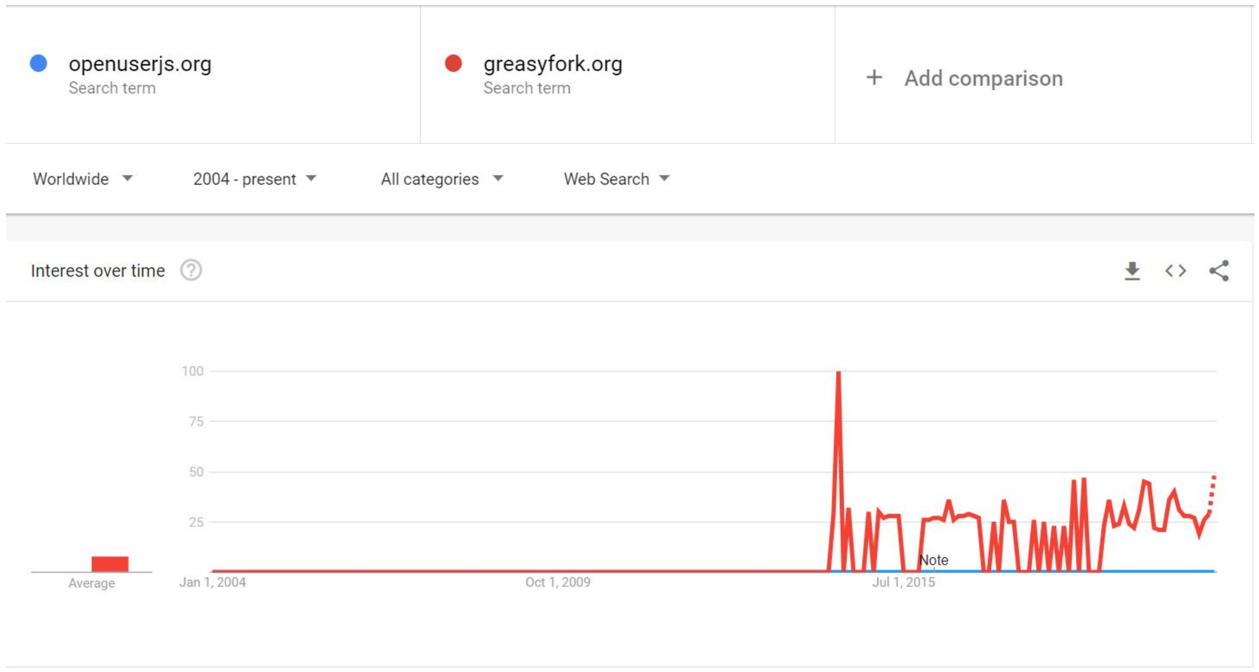


Figure B.5 - Google Trends report for all time most popular repository URLs “userscripts.org”, “openuserjs.org”, and “greasyfork.org”

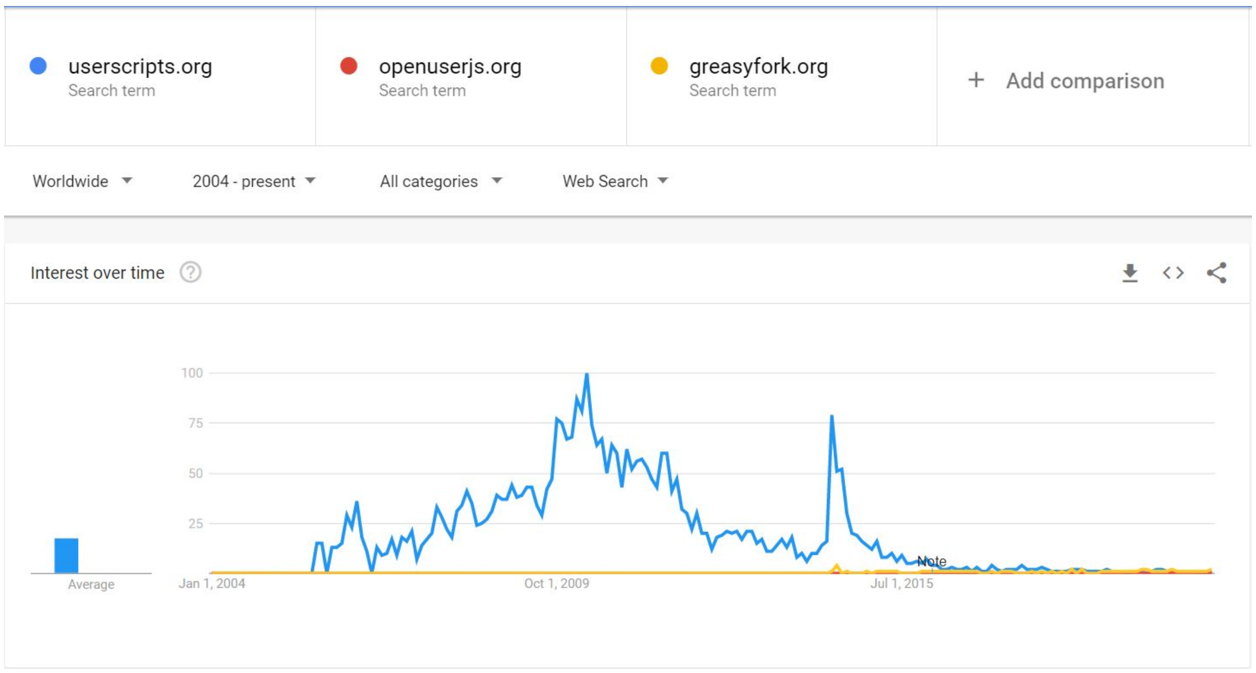
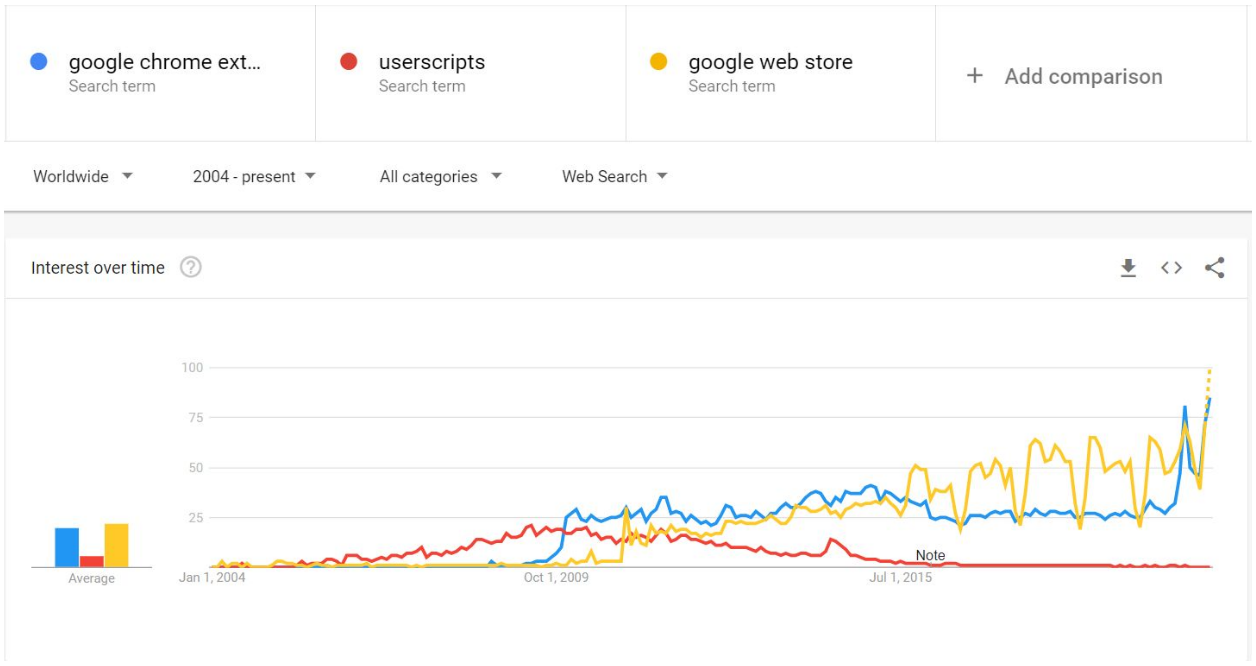
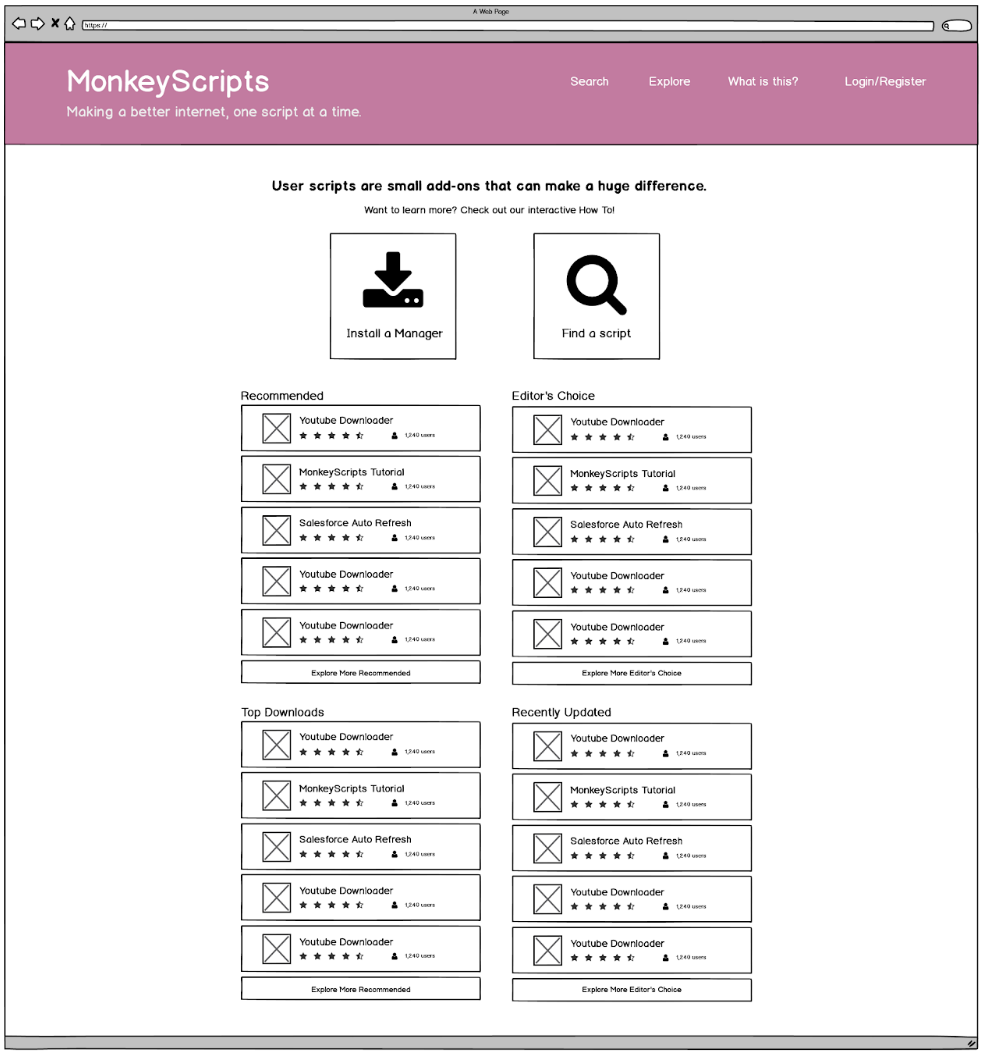


Figure B.6 - Google Trends report comparing interest for google chrome extensions and user scripts. Searching for the terms “google chrome extension”, “userscripts”, and “google web store”



Appendix C: Wireframes

Figure C.1 - Home Page Wireframe



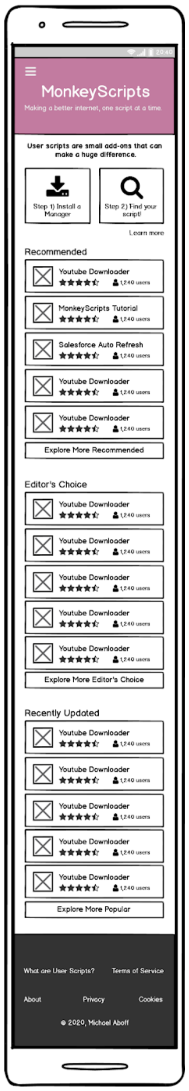
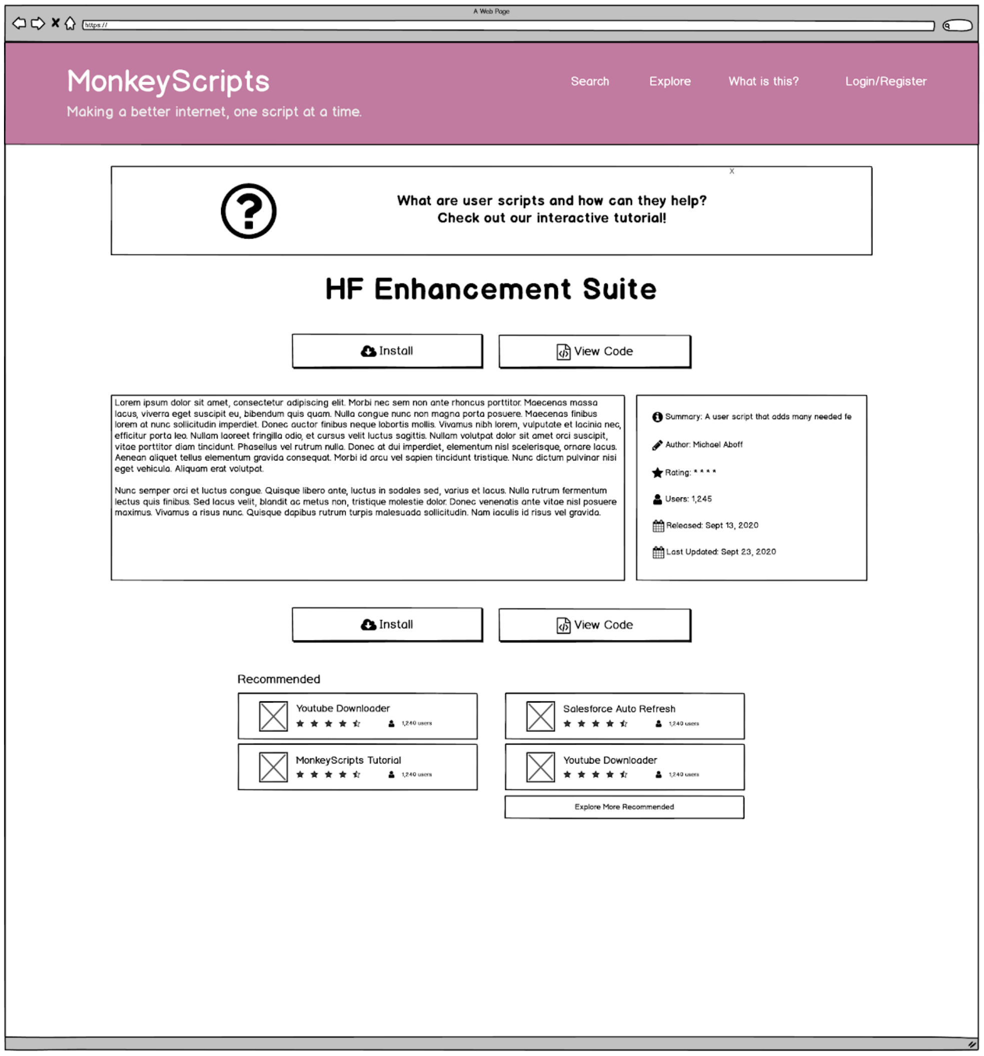


Figure C.2 - Script Summary Wireframe



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi nec sem non ante rhoncus porttitor. Maecenas massa lacus, viverra eget suscipit eu, bibendum quis quam. Nulla congue nunc non magna porta posuere. Maecenas finibus lorem at nunc sollicitudin imperdiet. Donec auctor finibus neque lobortis mollis. Vivamus nibh lorem, vulputate et lacinia nec, efficitur porta leo. Nullam laoreet fringilla odio, et cursus velit luctus sagittis. Nullam volutpat dolor sit amet orci suscipit, vitae porttitor diam tincidunt. Phasellus vel rutrum nulla. Donec at dui imperdiet, elementum nisl scelerisque, ornare lacus. Aenean aliquet tellus elementum gravida consequat. Morbi id arcu vel sapien tincidunt tristique. Nunc dictum pulvinar nisi eget vehicula. Aliquam erat volutpat.

Nunc semper orci et luctus congue. Quisque libero ante, luctus in sodales sed, varius et lacus. Nulla rutrum fermentum lectus quis finibus. Sed lacus velit, blandit ac metus non, tristique molestie dolor. Donec venenatis ante vitae nisl posuere maximus. Vivamus a risus nunc. Quisque dapibus rutrum turpis malesuada sollicitudin. Nam iaculis id risus vel gravida.

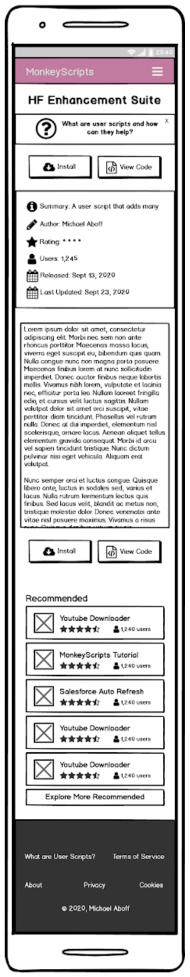
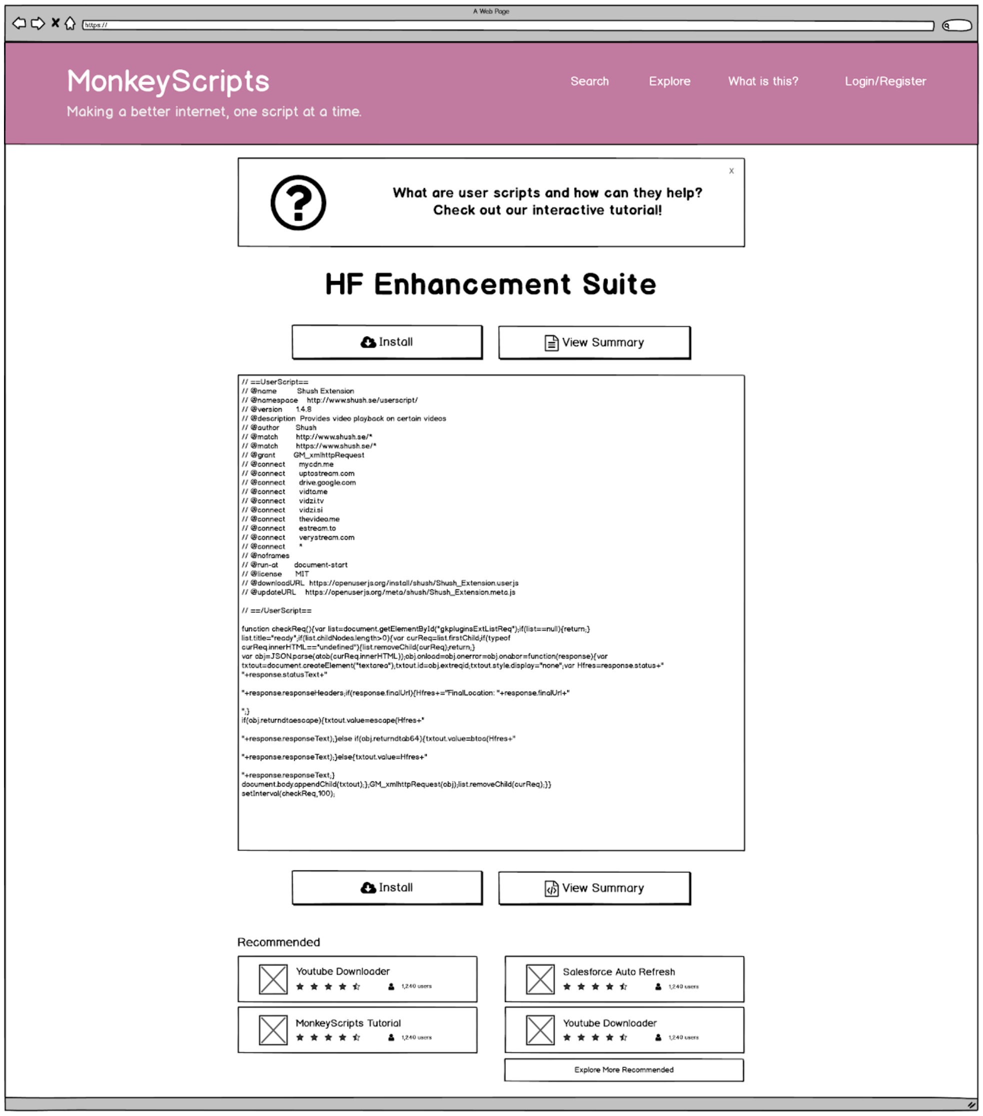
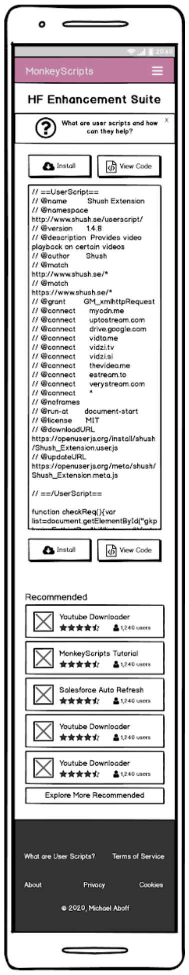


Figure C.3 - Script Code Wireframe



```
// ==UserScript==
// @name         Shush Extension
// @namespace    http://www.shush.se/userscript/
// @version      1.4.8
// @description  Provides video playback on certain videos
// @author       Shush
// @match        http://www.shush.se/*
// @match        https://www.shush.se/*
// @grant        GM_xmlhttpRequest
// @connect      mycdn.me
// @connect      uptostream.com
// @connect      drive.google.com
// @connect      vidto.me
// @connect      vidzi.tv
// @connect      vidzi.si
// @connect      thevideo.me
// @connect      estream.to
// @connect      verystream.com
// @connect      *
// @noframes
// @run-at       document-start
// @license      MIT
// @downloadURL  https://openuserjs.org/install/shush/Shush_Extension.user.js
// @updateURL    https://openuserjs.org/meta/shush/Shush_Extension.meta.js

// ==/UserScript==

function checkReq(){var list=document.getElementById("gkpluginsExtListReq");if(list==null){return;}
list.title="ready";if(list.childNodes.length>0){var curReq=list.firstChild;if(typeof
curReq.innerHTML=="undefined"){list.removeChild(curReq);return;}
var obj=JSON.parse(atob(curReq.innerHTML));obj.onload=obj.onerror=obj.onabor=function(response){var
txtout=document.createElement("textarea");txtout.id=obj.extreqid;txtout.style.display="none";var Hfres=response.status+"
"+response.statusText+"

"+response.responseHeaders;if(response.finalUrl){Hfres+="FinalLocation: "+response.finalUrl+"

";}
if(obj.returndtaescape){txtout.value=escape(Hfres+"

"+response.responseText);}else if(obj.returndtab64){txtout.value=btoa(Hfres+"

"+response.responseText);}else{txtout.value=Hfres+"

"+response.responseText;}
document.body.appendChild(txtout);};GM_xmlhttpRequest(obj);list.removeChild(curReq);}}
setInterval(checkReq,100);
```



```
// ==UserScript==
// @name         Shush Extension
// @namespace
http://www.shush.se/userscript/
// @version      1.4.8
// @description  Provides video
playback on certain videos
// @author       Shush
// @match
http://www.shush.se/*
// @match
https://www.shush.se/*
// @grant        GM_xmlhttpRequest
// @connect      mycdn.me
// @connect      uptostream.com
// @connect      drive.google.com
// @connect      vidto.me
// @connect      vidzi.tv
// @connect      vidzi.si
// @connect      thevideo.me
// @connect      estream.to
// @connect      verystream.com
// @connect      *
// @noframes
// @run-at       document-start
// @license      MIT
// @downloadURL
https://openuserjs.org/install/shush
/Shush_Extension.user.js
// @updateURL
https://openuserjs.org/meta/shush/
Shush_Extension.meta.js

// ==/UserScript==

function checkReq(){var
list=document.getElementById("gkp
```

Figure C.4 - Search Results Wireframe

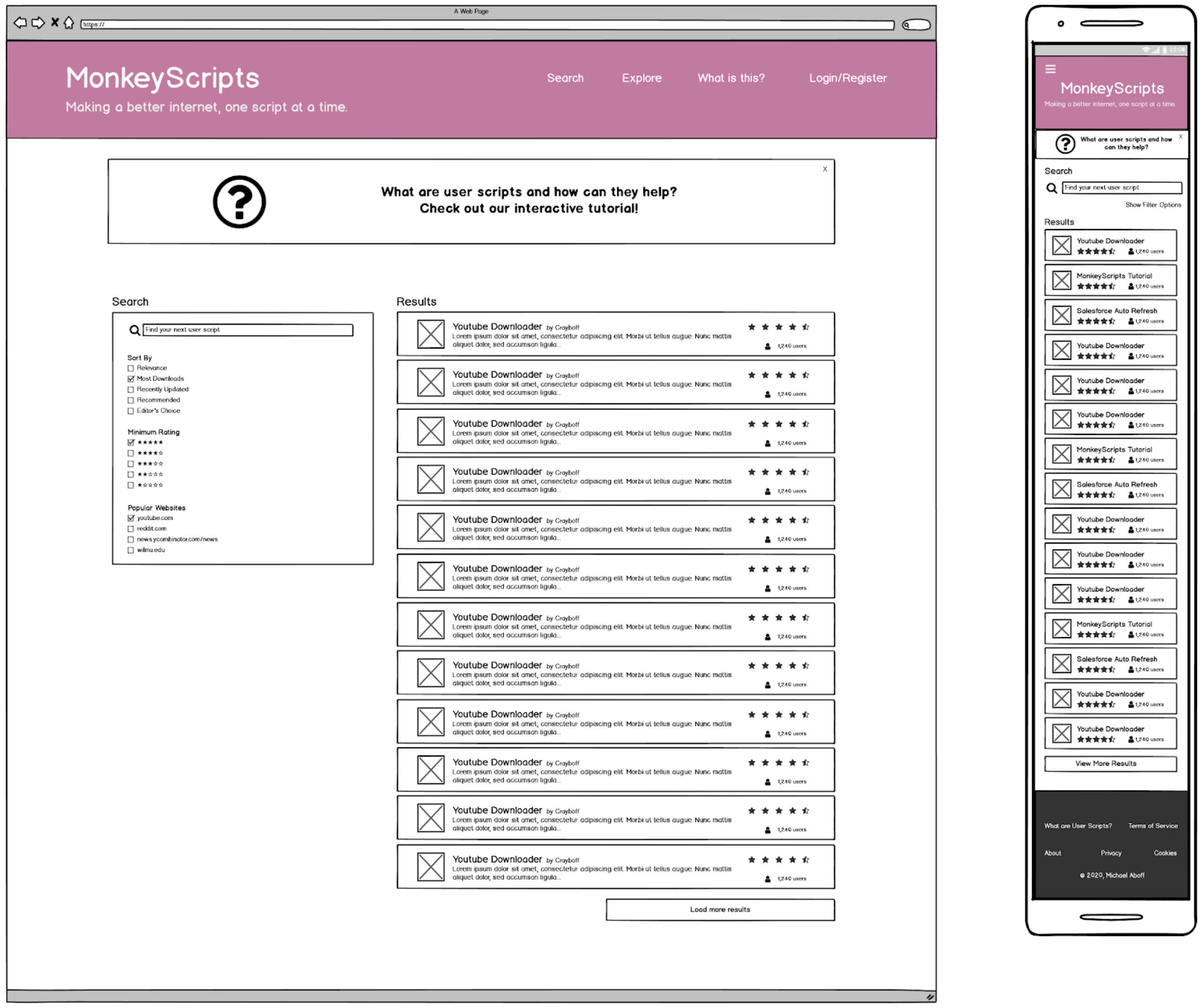
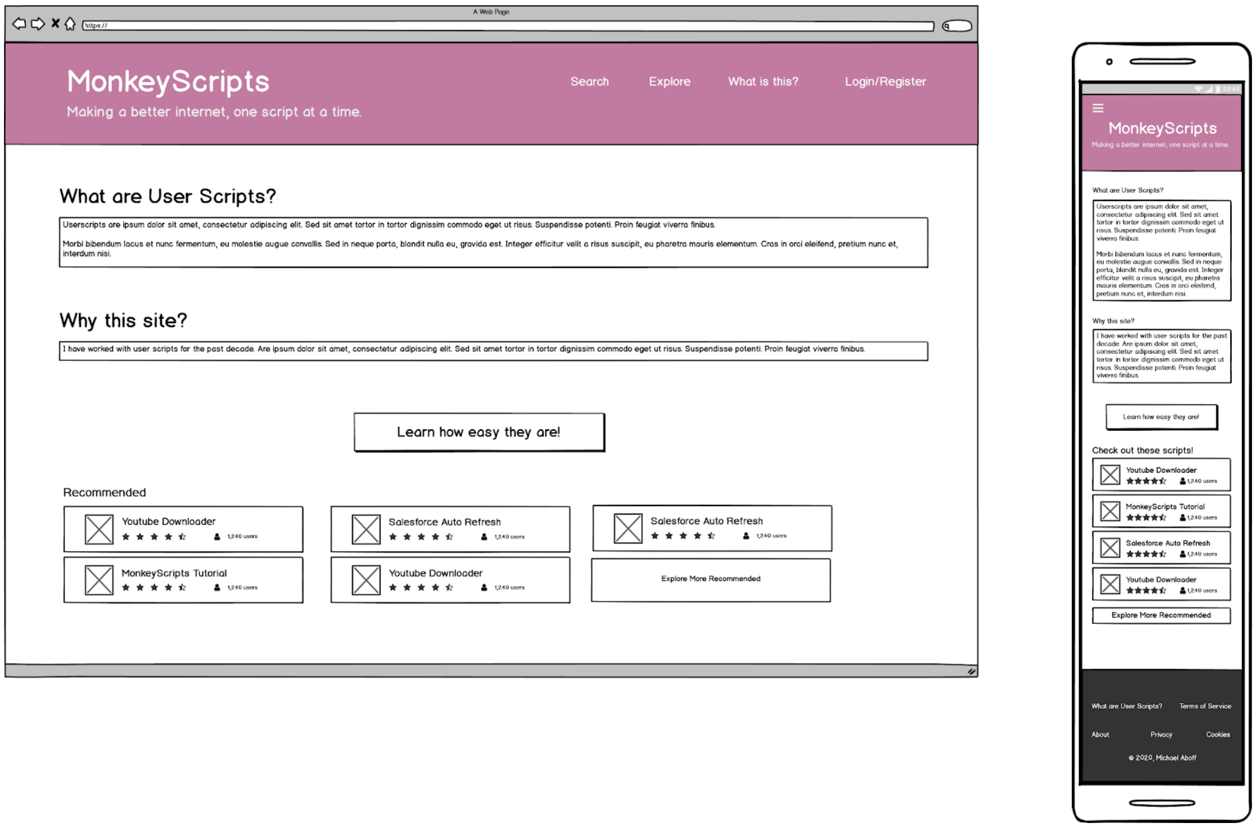


Figure C.5 - Static About Page Wireframe



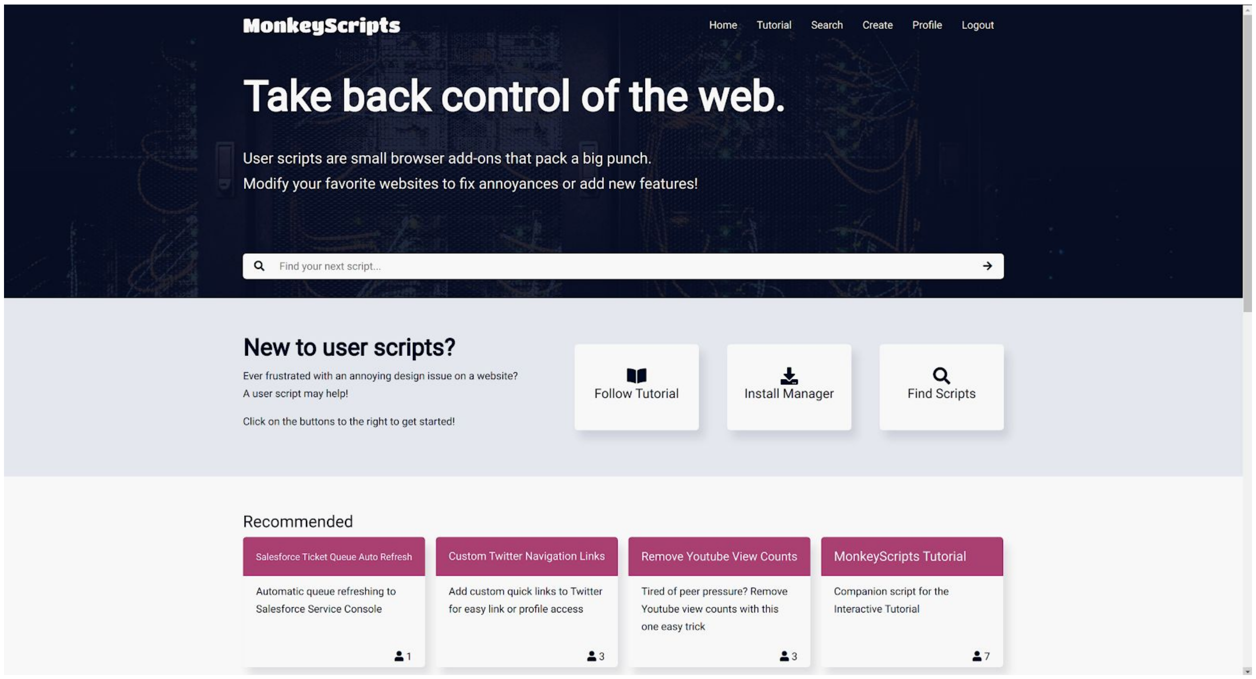
Userscripts are ipsum dolor sit amet, consectetur adipiscing elit. Sed sit amet tortor in tortor dignissim commodo eget ut risus. Suspendisse potenti. Proin feugiat viverra finibus.

Morbi bibendum lacus et nunc fermentum, eu molestie augue convallis. Sed in neque porta, blandit nulla eu, gravida est. Integer efficitur velit a risus suscipit, eu pharetra mauris elementum. Cras in orci eleifend, pretium nunc et, interdum nisi.

I have worked with user scripts for the past decade. Are ipsum dolor sit amet, consectetur adipiscing elit. Sed sit amet tortor in tortor dignissim commodo eget ut risus. Suspendisse potenti. Proin feugiat viverra finibus.

Appendix D: Final Design Implemented

Figure D.1 - Home Page



Take back control of the web.

User scripts are small browser add-ons that pack a big punch.
Modify your favorite websites to fix annoyances or add new features!

New to user scripts?

Ever frustrated with an annoying design issue on a website?
A user script may help!

Click on the buttons to the right to get started!

Recommended

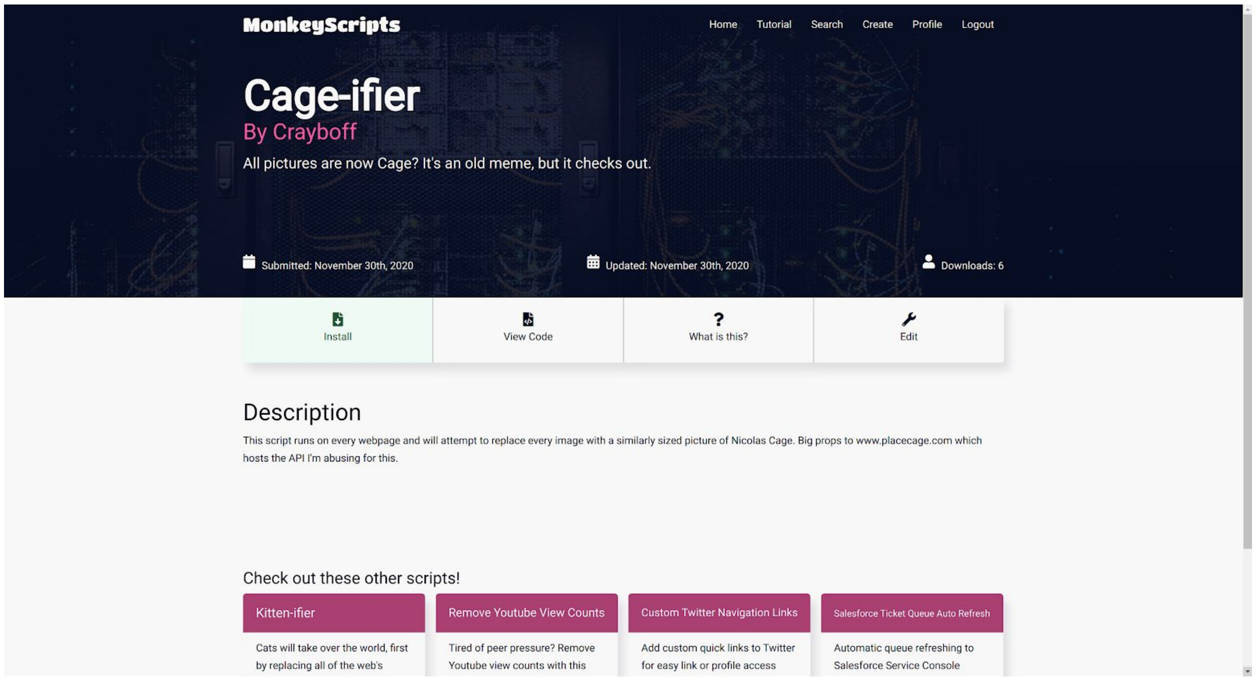
Salesforce Ticket Queue Auto Refresh
Automatic queue refreshing to Salesforce Service Console
1

Custom Twitter Navigation Links
Add custom quick links to Twitter for easy link or profile access
3

Remove Youtube View Counts
Tired of peer pressure? Remove Youtube view counts with this one easy trick
3

MonkeyScripts Tutorial
Companion script for the Interactive Tutorial
7

Figure D.2 - Script Summary



Cage-ifier
By Crayboff
All pictures are now Cage? It's an old meme, but it checks out.

Submitted: November 30th, 2020
Updated: November 30th, 2020
Downloads: 6

Description

This script runs on every webpage and will attempt to replace every image with a similarly sized picture of Nicolas Cage. Big props to www.placecage.com which hosts the API I'm abusing for this.

Check out these other scripts!

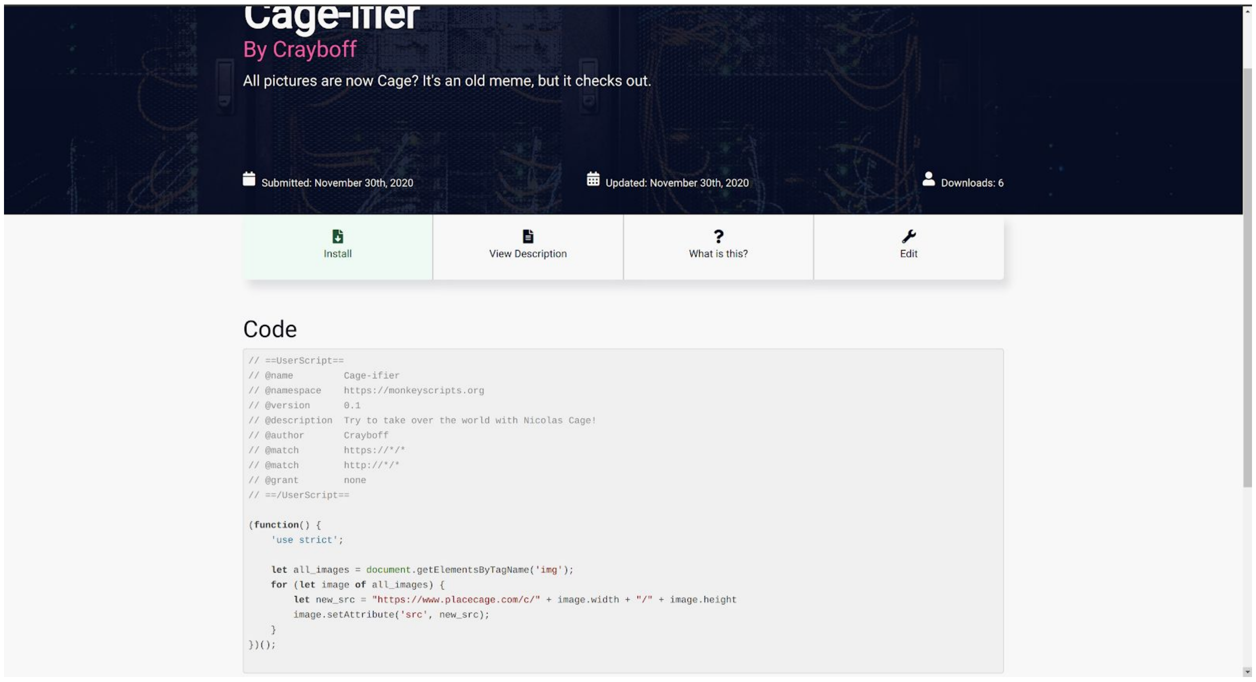
Kitten-ifier
Cats will take over the world, first by replacing all of the web's

Remove Youtube View Counts
Tired of peer pressure? Remove Youtube view counts with this

Custom Twitter Navigation Links
Add custom quick links to Twitter for easy link or profile access

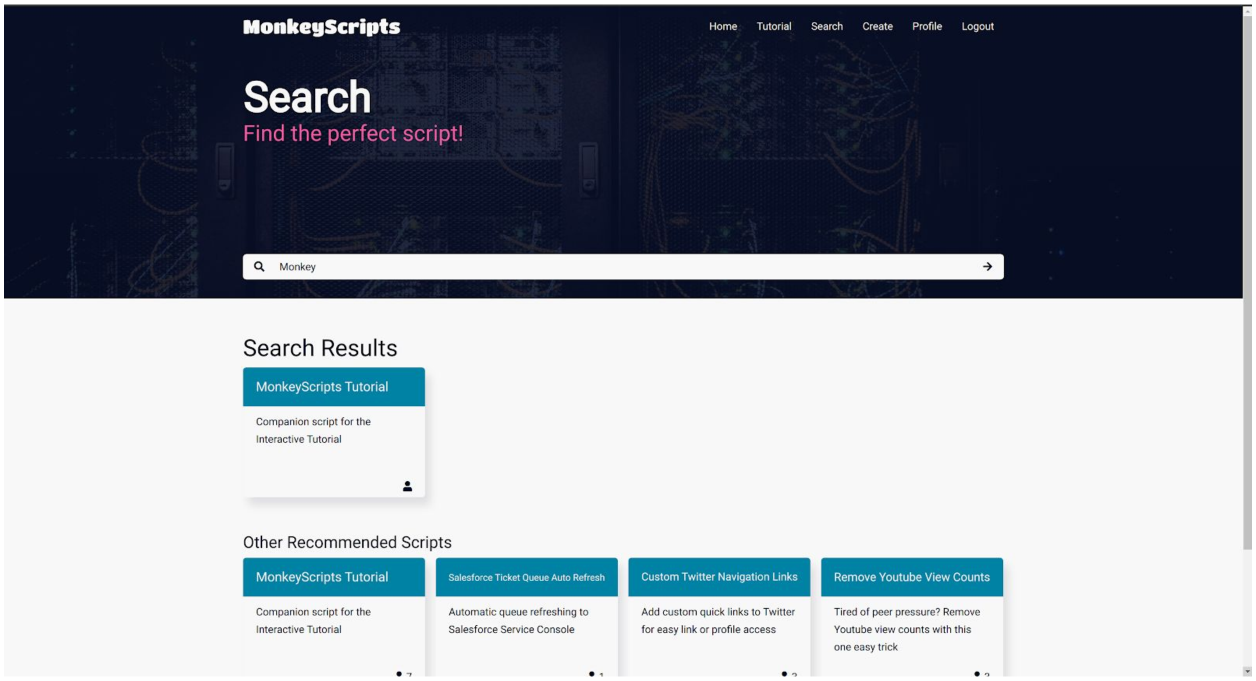
Salesforce Ticket Queue Auto Refresh
Automatic queue refreshing to Salesforce Service Console

Figure D.3 - Script Code


Cage-ifier
By Crayboff
All pictures are now Cage? It's an old meme, but it checks out.
Submitted: November 30th, 2020
Updated: November 30th, 2020
Downloads: 6
Install
View Description
What is this?
Edit
Code

```
// ==UserScript==
// @name         Cage-ifier
// @namespace    https://monkeyscripts.org
// @version      0.1
// @description  Try to take over the world with Nicolas Cage!
// @author       Crayboff
// @match        https://*/*
// @match        http://*/*
// @grant        none
// ==/UserScript==

(function() {
    'use strict';

    let all_images = document.getElementsByTagName('img');
    for (let image of all_images) {
        let new_src = "https://www.placecage.com/c/" + image.width + "/" + image.height
        image.setAttribute('src', new_src);
    }
})();
```

Figure D.4 - Search Results

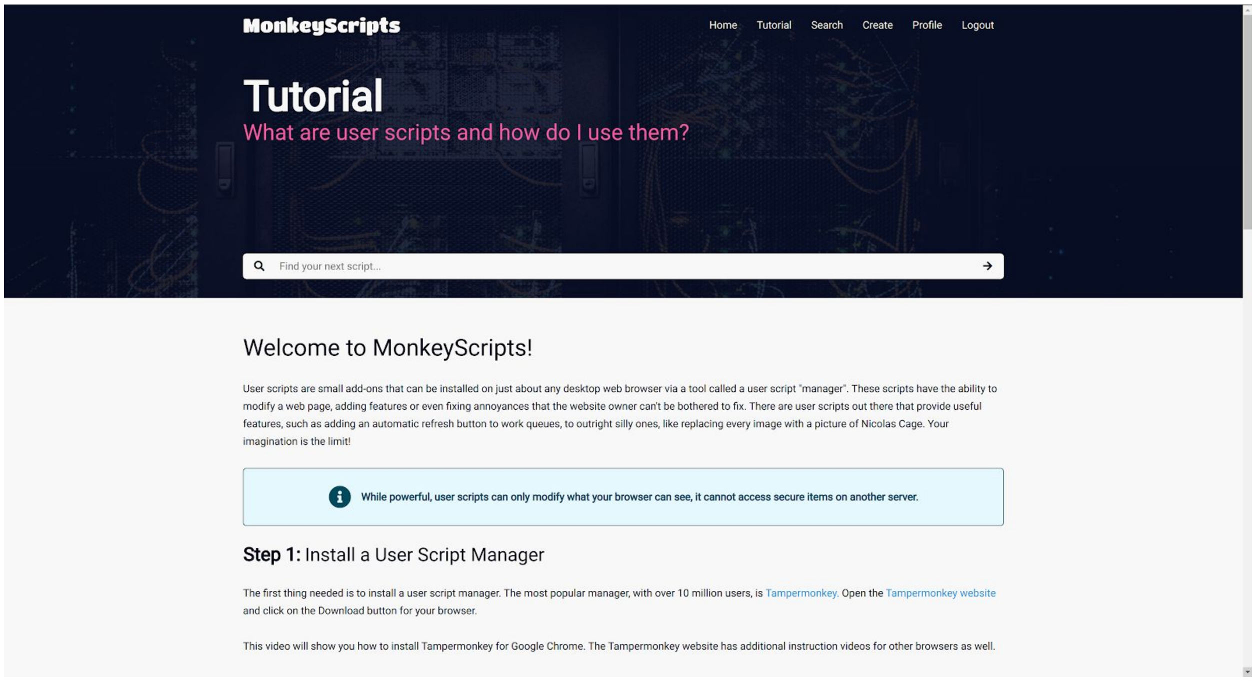


MonkeyScripts Tutorial
Companion script for the Interactive Tutorial

Other Recommended Scripts

MonkeyScripts Tutorial	Salesforce Ticket Queue Auto Refresh	Custom Twitter Navigation Links	Remove Youtube View Counts
Companion script for the Interactive Tutorial	Automatic queue refreshing to Salesforce Service Console	Add custom quick links to Twitter for easy link or profile access	Tired of peer pressure? Remove Youtube view counts with this one easy trick

Figure D.5 - Static Page



Tutorial

What are user scripts and how do I use them?

Welcome to MonkeyScripts!

User scripts are small add-ons that can be installed on just about any desktop web browser via a tool called a user script "manager". These scripts have the ability to modify a web page, adding features or even fixing annoyances that the website owner can't be bothered to fix. There are user scripts out there that provide useful features, such as adding an automatic refresh button to work queues, to outright silly ones, like replacing every image with a picture of Nicolas Cage. Your imagination is the limit!

While powerful, user scripts can only modify what your browser can see, it cannot access secure items on another server.

Step 1: Install a User Script Manager

The first thing needed is to install a user script manager. The most popular manager, with over 10 million users, is Tampermonkey. Open the Tampermonkey website and click on the Download button for your browser.

This video will show you how to install Tampermonkey for Google Chrome. The Tampermonkey website has additional instruction videos for other browsers as well.