Software Development and Market Research Process of Plasma Software Distribution

Connor Moore

Murray State University Honors College

HONORS THESIS

Certificate of Approval

Plasma Software Distribution Development

Connor Moore

May 2024

Approved to fulfill the requirements of HON 437	________________________________ Dr. Jason Owen, Professor Computer Science and Information Systems
Approved to fulfill the Honors Thesis requirement of the Murray State Honors Diploma	________________________________ Dr. Warren Edminster, Executive Director Honors College

Examination Approval Page

Author: Connor Moore

Project Title: Plasma Software Distribution Development

Department: Computer Science and Information Systems

Date of Defense: 4/26/2024

Approval by Examining Committee:

(Dr. Jason Owen, Advisor)	(Date)
(Dr. Victor Raj, Committee Member)	(Date)
(Dr. Matthew Tennyson, Committee Member)	(Date)

Plasma Software Distribution Development

Submitted in partial fulfillment
of the requirements
for the Murray State University Honors Diploma

Connor Moore

April 2024

Abstract

When trying to find the right software for scientific research, one may have to comb through the internet to acquire a suitable tool. Much of the scientific software online is in mostly unknown web pages where the only way to find the software is to already know about it, be told about it, or find the software by pure chance. Even worse, with few verification systems in place they may try a new program only for it to turn out to be malware. The search for the right software takes time away from the scientists, slowing the overall pace of scientific innovation. Plasma Software Distribution, a web application, intends to remove much of the busywork of finding the software, allowing scientists to do more of the work that improves lives.

Table of Contents

1 INTRODUCTION

Scientific research and technological production go hand in hand, and perhaps no more so than in the production of technology to assist in scientific research. However, finding the right technology, specifically software, for one's research can be a daunting task. The time spent looking for programs and researching the best ones to use can slow progress. Since there are no services that group the programs together, the reviews for each program, if there even are any reviews, are affected by bias. This makes finding the right program even more difficult, as one cannot trust the reviews for the programs. There is a risk that better programs for researchers to use in their fields exist just out of sight on the internet, and without a way for them to be brought to a scientist's attention the scientist can use the software.

An application to assist in finding the right tools for research and scientific pursuits could not only further science, but also has the potential to be profitable. Plasma Software Distribution intends to solve the issue of how to find the right programs. To solve this issue, research was conducted to develop Plasma into the premier way to acquire software in scientific fields.

Plasma Software Distribution will compile and store scientific programs on an online marketplace, allowing researchers, professors, and other academics to find the software they need with little effort. Additionally, Plasma will support a job auction board where those academics can hire freelancers to create the software they need in case the program does not exist in the marketplace. This will help solo researchers, freelance programmers, and universities.

1.1 Problems to Solve

To solve the issue of finding the right programs, one needs to understand the components of finding the right software. First, there must be a way to search for the software. Not only should the search engine be able to find the software, but it should not give unrelated results. Second, the software must exist. This means that not only should there be software that does what the searcher wants, but also that the software must be capable and high-quality. Therefore, the user needs an excellent search system and the right tools and programs available for the search engine to find. Thus, the two most important questions to answer are these: How can one provide a top-notch program searching system, and how can one store high-end programs?

1.2 Hypothetical Program to Solve Problems

A theoretical program, which will be named Plasma Software Distribution, could solve both issues discussed with one feature, a software marketplace. A properly developed software marketplace, that being one with both a capable software search system and containing quality programs, would significantly ease the troubles of finding the right software. Before starting development on those two features, success for both areas must be defined. Success for a search engine is where a user can know generally what they are looking for, type related terms or a description of the object, and then find the object. As for success for high quality software, that is when one can download software and not have to worry about the downloaded program being a bug-filled mess, malware, or doing something entirely different than described. An application that has found success in both of those features will solve the issue of finding the right software.

2 RESEARCH

To ascertain the best method of solving these issues, market research was conducted to find what abilities Plasma Software Distribution would have that would be most helpful for the consumer. This research included emailing university professors for interviews, reading existing literature on the subject, and studying similar services. The most helpful portion of the market research was also one of the smallest, that being the interviews. Cold emailing professors was not an effective way to get interviews, and only one professor responded with interest. Thankfully, Dr. Wesley Totten, Chair of the Department of Agriculture, Geosciences, and Natural Resources at UT Martin had excellent responses and information regarding how a scientific software repository would be useful to him.

2.1 Interview

During the interview, Dr. Totten talked about some of the software he used in his work. This list of software included statistical software, some software for engineering, and then mostly teaching software. He also said that he uses the software from two to three hours a day maximum. Although he does not constantly use software, he does use it enough for it to be important. Furthermore, he and his department do pay some money for software, sometimes up to $5000, but sometimes they get it for free. His main annoyance with finding software is that they need it to be user and student friendly. As for features he would like to see in a software repository, he was interested in searching by specific terms, such as landscape design, and by field, such as Agriculture Engineering or Plant Science.

Regarding acquiring software, Dr. Totten said that finding the right software can be difficult. Additionally, he says he spends about an hour a week looking for software. While this is not a particularly long time, eliminating this hour as well as making sure the time spent searching is even more fruitful will help. However, due to university licensing, Dr. Totten does not necessarily get to change what software he uses even if he finds one that is better. For personal use, though, whether one changes software can depend on their discipline. In fact, changing software can be absolutely worth it for some. Finally, difficulty installing a program can affect whether one wants to use the program although if the program is excellent, it may be worth the trouble.

Dr. Totten was especially interested in the idea of a job auction to create bespoke software. Such an auction would allow users to hire freelancers to make software to their specifications. He said it would be extremely useful for most disciplines, which means that the marketplace's auction system needs to be top notch. Of course, hosting so many programs, profiles, and general data comes with a cost. Since the price of hosting a software repository will only grow with the number of programs and users, proper planning must be done to determine how much money users are willing to pay for the service compared to how much the service will cost. Dr. Totten said that monthly they could pay up to $250 or $350, but only with grants. For a single department, they may be able to pay $100 for the professors to use the software. Assuming that this single data point applies to more universities than just UT Martin, $100 a month may not be enough to host the data, not even counting paying employees. To acquire proper financial backing, there would have to have a massive number of universities using and paying for the software and hopefully outscale the operating costs. Outscaling the operating costs is possible, but making Plasma Software Distribution profitable may be a challenge and finding additional sources of income from the web application will be a priority.

2.2 Literature

Georgia Tech's Partnership for an Advanced Computing Environment, or PACE, team created and maintains a software repository for Georgia Tech (PACE). Their work on their software repository is strikingly similar to Plasma, including the limited manpower. To get the most out of their limited numbers, they used a 3-tier system to find what software is most important to their users and then they focused on heavily supporting that software (Belgin et al). Software that was slightly less important and used but still worth caring for had less support, but they would help with installation and critical updates. All other software they left to the community to maintain. Their approach allowed them to provide active and high-quality support to the majority of their users with no more than thirteen team members. Although software and technical support for the stored software is not a planned feature of the planned software repository, their method of prioritizing software by gathering data on how often the programs are used is an excellent example of how to manage a large database with a limited staff.

Philip A. Bernstein and Umeshwar Dayal define a repository in their work "An Overview of Repository Technology" as "a shared database of information about engineered artifacts produced or used by an enterprise." These artifacts are anything that can be stored digitally, from video games to blueprints. Therefore, a software repository is a digital repository that stores software. There are various functions a software repository can have, such as checking out or checking in code, a version control system, and a configuration control system (Bernstein, 706). Plasma contains elements of each of these parts of a software repository. In the case of checking in or out source code, uploaders will submit or retrieve the compiled, ready-to-run program. However, a version control system will be quite useful for Plasma, although expensive. Each version of a program could be stored, but that would be too much data for too little benefit. Instead, only versions of programs before possible breaking changes would be stored. That way, Plasma does not need as much space and any user that needs a program to continue working can be certain that there is a version that works for them. Finally, the configuration control is for the various requirements software can have in relation to other software. So long as the program lists the other programs and pieces of software it requires, or preferably works without any necessary manual configuration, then a configuration control system would be simple to create.

Due to the massive collection of programs contained in software repositories, there are unique opportunities to gather further information on the software contained. This is known as mining a software repository. Ahmed Hassan explains in his article “The road ahead for Mining Software Repositories” the various ways one can gain information from the stored programs and software in a repository, as well as achievements in the field of Mining Software Repositories. Reading the article sparked a few ideas on how Plasma could improve itself from gathering data. Although mining a software repository in a more traditional way is difficult for Plasma because Plasma’s goal is to store the compiled code, there are still opportunities to gather data. For instance, one could store what programs are popular, what scientific fields use the most technology, where gaps exist in software coverage, and what kinds of programs are most requested in the job auction. All of this data can be used to improve Plasma’s ability to serve its users, but the data could also be monetized. There are two ways to monetize data, implicitly and explicitly (Shukla et al). To monetize the data implicitly, Plasma would be using the information gathered to improve its service for the user or reduce costs. For instance, if Plasma knows about what programs are popular or necessary, it could feature specific programs that might be useful to each field. Furthermore, Plasma could recommend different paths for freelancers so that they can learn what they should focus on in the job auction board, such as what languages work best or what features do consumers usually want. However, Plasma could also explicitly monetize the gathered information by selling the data to other companies. There are some ethical considerations and rules to be made before any actual monetization happens, though.

First, Plasma should not and will not make any money off of the actual content of a program without permission. All Plasma can do is gather information about the access to the program, nothing more. Second, Plasma should not and will not make any money off of personal information about users, such as the country they live or what field they belong to. Finally, Plasma should not and will not sell data to companies that have malicious intent. With those considerations in mind, the ability to sell information to advertisers or software development companies would not only improve outreach and knowledge about the programs Plasma contains, but also improve knowledge about what programs need to be made to help scientific innovation.

Because of the freedom users would have to upload software, proper malware detection techniques are required for user safety. In “A Comprehensive Review on Malware Detection Approaches,” the authors Aslan and Samet list a variety of ways to detect malware. Although they say there is no one way to detect all malware, many of the listed methods would work well for Plasma Software Distribution. The main ways that are listed that Plasma will use are signature-based detection, behavior-based detection, and cloud-based detection. To summarize the three, signature-based detection revolves around having an existing database of malicious code. Next, behavior-based detection detects malware by tracking how it acts in a system. Using a virtual machine and proper precautions, one can run possibly malicious code safely to check if it truly malware. Finally, cloud-based detection uses different methods to detect malware, but instead of detecting malware on the machine one can send a file over the internet for inspection.

All three of these methods can work for Plasma, but each has their own pros and cons. All together, however, these methods can provide greater safety for users.

Since Plasma Software Distribution is hosting so many programs, any breach of security could cause major damage by leaking these programs. Therefore, Plasma should make an effort in securing and encrypting the software it contains. Yang et al. discusses various previous ways to ensure data confidentiality, as well as what the most suitable method would be for cloud services. Given Plasma's use over the cloud, finding the best encryption technique for software on the cloud is key to providing a safe and secure service. The information contained in the article provides useful insights for what Plasma should focus on when developing or looking for its software encryption.

2.3 Steam

Beyond interviews and literature reviews, there were also similar services to study. Although none served exactly the same purpose as Plasma, many fulfilled comparable roles. Perhaps the greatest inspiration for this project, Steam, does the same thing Plasma Software Distribution does except for video games and video game related software. Created by Valve Software, Steam is a free platform that takes a cut of purchases made on its marketplace. Being a multimillion-dollar application, Steam is highly optimized for its users, so studying how it operates and why it does what it does can assist with Plasma's user experience. Of particular note is how Steam organizes its store as well as various pages. Users access access the main sections from anywhere else in the application from an overhead menu and from there one can dive deeper into the marketplace, personal profile page, their own software library, or the community page. While there is currently little reason to have a community page in Plasma, how Steam designed their marketplace, owned software library, and profile page is worth drawing significant inspiration from.

2.4 Fiverr

Fiverr is a job auction service which functions similarly to Plasma Software Distribution’s job auction service. However, there are some significant differences in planning between Fiverr and Plasma. Fiverr has its freelancers post what they do and then someone can hire the freelancer based on the job. A key difference is that Plasma will be posting the jobs that need to be done and then freelancers can apply to the job with a streamlined application system. In this way, Plasma is also similar to job application sites like Indeed, Dice, or even the job application portion of LinkedIn.

There are also organizations that focus on academic software repositories, much like Plasma, but their focus is more limited. Georgia Tech has its PACE team which stores, distributes, and provides help and support to the software used at Georgia Tech. Additionally, the European Science Cluster of Astronomy & Particle Physics ESFRI Research Infrastructure, or ESCAPE, has the Open-source scientific Software and Service Repository, or OSSR, for much of the same functions as Plasma. However, ESCAPE’s OSSR is for data analysis, astronomy, and particle physics, which is not as general as the goal for Plasma. However, since these both fit with Plasma’s main goal but on a smaller scale, once Plasma Software Distribution gets large enough, collaboration between Plasma, ESCAPE, and PACE may be worth looking into.

3 METHODS OF CREATING PLASMA SOFTWARE DISTRIBUTION

With the theoretical program and the research conducted in mind, one can begin to find the best way to go about creating the best service possible. Plasma Software Distribution will utilize third party-APIs due to their quality and the limited team of developers. These third-party tools are independently maintained, thus allowing Plasma to have excellent functionality without the need to maintain possibly delicate infrastructure. Additionally, Plasma uses various computer languages in its construction.

The base of Plasma Software Distribution is in HTML, CSS, and JavaScript. For the frontend, the team used React and Bootstrap. React is a JavaScript Library that contains many useful components for building an interactive web application's frontend. The team was experienced with React, so it was the obvious choice to start building Plasma. The team used it all across the code, but some of the better uses of React are its useStates which allowed the program to more easily track and change variables and states.

Bootstrap is very similar but has been implemented mostly for the user interface. In particular, Bootstrap's buttons are clean and functional. Plasma's programmers have used Bootstrap previously as well, and the team's frontend engineer likes it. Of course, to run JavaScript on the backend there must be a runtime environment, so the team decided on Node.js for much of the same reasons as React and Bootstrap. The team has used Node.js before, and it served the required purposes. Finally, npm was used for package installation and dependencies due to previous experience and ease of use.

3.1 IDE and Version Control

Beyond the basics for making a web application, there are some more general pieces of software that the Plasma team utilized. First, they used GitHub for both version control and as a code repository. This was useful to them so that they could share code among the team and keep the application up to date. Furthermore, the team chose GitHub because it is the industry standard and the programmers all had a fair amount of experience with it. Although the choice of Integrated Development Environment, or IDE, could change from team member to team member, the team used VSCode for the bulk of the programming. Although some members were not as experienced with VSCode as IntelliJ, they switched in the middle of the project because VSCode worked better with JavaScript and had a number of useful add-on and features. Namely, IntelliJ failed to color the code, while VSCode had that feature. The code changing its color depending on what it was significantly helped readability, so this was a necessary change.

3.2 Third Party APIs

Beyond the necessary baseline software and the periphery useful development software, there are also third-party APIs that were extremely useful in Plasma Software Distribution’s development. The two features best implemented using these APIs are monetary transactions and searching. Both are very complicated and financial transactions are extremely important to get right, so using a reputable high-quality service allowed Plasma to ensure the security of users’ finances. For the payment API, Stripe’s extensive documentation and ability to handle recurring payments and subscriptions made it the clear choice. Stripe’s purpose as a payment tunnel is to connect the members of a financial transaction before any money is touched so that one can be sure that the money will not be intercepted or dropped. As for the searching software, Lucene fits Plasma’s needs almost exactly. Lucene can be implemented to search by program name, search the program description, the type of field the program belongs to, and is free and open source. The only issue with it is that it is in Java, not JavaScript. The team may have to work on adapting its capabilities to Plasma, but the reward is well worth the effort.

3.3 Backend Development Strategy

The final and perhaps most important third-party software is actually a whole suite of useful and related software. Since the Plasma Software Distribution team does not own their own server, they must rely on cloud services. For Plasma's cloud service, Microsoft Azure fulfills every need. While there were a variety of reasons to choose Azure, the main one was cost. Azure had a free trial and an excellent pay-as-you-go system allowing Plasma to only be as costly as it needs to be. Furthermore, Azure has a plethora of features to use, but the main feature Plasma uses is Azure's blob storage. Azure blobs are any sort of file one can store, from a photo to a video to a text file. Anything that is represented by ones and zeroes, which is everything on a computer, can be stored as a blob over the cloud with Azure blob storage. The team also uses it for hosting a SQL database to store the list programs and locations, as well as what is listed on their webpages. Additionally, SQL is used to store a list of users and what programs they have previously downloaded. Not only is Azure used for storage, but it is also used for hosting the website. Azure can link to the GitHub repository for Plasma so once the development build is complete the code can be released to production and Azure will automatically deploy the new version of the webpage. That is not all that Azure is used for though. Plasma also utilizes the cloud service for access management and security.

4 RESULTING PRODUCT

Although Plasma Software Distribution has yet to be finalized into a marketable product, the format of the initial product is clear. Plasma will have certain features and a clear idea of how a user will interact with the application. Additionally, the web application's security, monetization, and once the initial product is finished, possible future routes to take the web application are being accounted for.

4.1 Features

With the tools previously discussed, Plasma's features can be built. Besides the obvious marketplace, the interest Dr. Totten had in a job auction suggests that Plasma should focus on that as another main feature. Additionally, to help understand how the app should function for the two main types of users, one should consider the ideal scenarios for how a user should interact with the app.

One of Plasma Software Distribution's primary features is the software marketplace. This is where customers can search for the software that will meet and exceed their needs. Since each program will have its own detailed webpage, customers can be well informed about the program they want before they buy or download it. Not only that, but companies that upload the software will have a straightforward process due to the upload form. This form looks like a replica of the webpage so that the uploader knows what their prospective downloaders will see when they look at the page.

If the customer cannot find the software they want, they have other options. Plasma Software Distribution's other main feature is a job auction board where researchers, professors, and scientists can describe the software they want on a job page. Although the page looks similar to a software page on the marketplace, it serves a different purpose. The job page is there to provide a description of the needed software, as well as relevant literature and information. Then freelance programmers and possibly even companies can look at the software jobs available and apply to ones they believe they can produce. After that, the researcher can look at the various programmers and decide which one works best for them based on the time and money they say they require, as well as previous customer reviews. Next, the programmer will submit a timeline of when they expect to get the project done so that the researcher knows what the point the project should be on at any given date. Finally, the researcher pays the full price of the project, and the programmer gets a portion. The rest of the money is withheld for arbitration in case of issues between the researcher and the program.

4.2 Users

The users of Plasma Software Distribution generally fill two roles, uploaders and downloaders. Additionally, there is a third type of user account for Admins. Uploaders are the ones who upload programs to the marketplace or to clients on the job auction board. While they might not use Plasma for its software distribution capabilities, they will be providing the content that Plasma is distributing, making them vital to Plasma. These uploaders could be anything from a single freelancer to a whole company. The second type of user is the downloader. Most likely, these users will make up the bulk of the user base and are the ones who are getting the most functionality out of Plasma. These kinds of users are typically researchers, scientists, professors, or lab workers who want the best software for their job without having to search all across the internet for it. Therefore, they will use the marketplace and the job auction board to either acquire the software they need or pay for its creation. Finally, Admin accounts help the whole system run smoothly.

Each user will also have a profile page. This page will contain a list of their previously downloaded programs so that they can redownload ones they like and do not have to pay for the same program multiple times. The page will also contain a list of reviews, both reviews of the user and the user’s reviews of software. They may also include a personal description and organizations they may belong to.

4.3 Optimal User Experience for Uploaders and Downloaders

Because of the difference between uploaders and downloaders, there are different examples of the optimal user experience. For downloaders, an example would be a physics professor whose preferred software has been deprecated. Since this professor needs to find new software, they turn to an excellent new web application known as Plasma Software Distribution. Because the university they work for cares about the research the professors do, access to the marketplace and the ability to upload jobs to the job auction board is paid for already, so all the professor has to do is look for the right tools with Plasma's search system. The program that the professor used had two functions; the first function was to check how fast a certain particle can move in various conditions, and the other was to check how close these particles can naturally get to each other in similar conditions.

Due to the wide array of software Plasma stores, the professor can easily find a program that takes parameters relating to the particle and calculates the speed. Should the professor later have a problem with the program, they can go to a troubleshooting page linked to the specific program. There, admin accounts or accounts linked to the company or person who uploaded the program can help the professor solve their problem.

However, since the particle the professor is studying is so rare and the calculations for proximity happen to be specific to the particle in question, the professor cannot find a suitable program for proximity. Therefore, they go to the job auction board. On the board, they post a description of what they want the program to do, relevant literature relating to the particle, example input and output, and the preferred payment range and time to completion. Then, they receive many offers and listings on the board. They wind up choosing a computer science and physics dual major student based on their profile. Their experience, project timeline, payment required, and reviews are the best choice in this case. The professor gets the program they need and can continue to further scientific research and academic knowledge with the assistance of Plasma Software Distribution.

As for downloaders, an example of an optimal user would be a student of computer science who wants some more work experience and money. This student in particular happens to be dual majoring in physics, so they go to the job auction board to find programming work relating to physics. The well-designed search feature makes it easy for them to locate a job from a particle physicist to make a program that can calculate the minimum proximity a group of specific particles can have in various conditions. Although the student has not worked with this particle before, the physicist has described the program and linked enough literature that the student is confident that they can create the program. Because of the student’s past experiences with programming for Plasma, they know about how long making the program will take and how much money they should ask for their services. They create a timeline for their project and describe past projects in their auction posting and eventually get chosen for the job. Then, they receive a portion of the payment and get to work. Thankfully, they manage to stick to their timeline, deliver the project with no unmanageable complications, and receive the rest of their payment.

Although the given examples were a professor and a student, there are many other possible users for Plasma. Researchers can use Plasma for their innovations and lab managers can use Plasma for a common suite of software so that all lab technicians can easily find and download the right tools. Students can use Plasma in their classes to find the software their professors require them to use and professors can use Plasma to assign software for their students to use on homework. Finally, freelance programmers can use Plasma as a source of income.

As mentioned earlier, there is a third type of account other than uploaders and downloaders. Admin accounts would have special permissions so they can properly maintain Plasma. Since maintenance and support for the individual programs comes down to the program's developers, admin accounts would only be responsible for tasks such as adjudication, bug fixing for Plasma itself, and testing. These accounts would most likely only be active while the user is working, and have the ability to download and use programs to test them without paying, as well as not having to pay to access Plasma.

4.4 Security Measures

Since Plasma Software Distribution is hosting other companies' software, security is a top concern for both the companies uploading software and the users downloading the programs. Plasma can work off a whitelist system for what companies are allowed to upload software. While this may take time at first, having this whitelist opens up more opportunities in the future. Additionally, since a company must be researched before added to the whitelist, there is much less likelihood of foul play. However, in case a whitelisted company does decide to upload malicious code with their program, Plasma will check the code against a malware database before allowing the program on the store. If the program contains elements in the database that signify that the program may be malicious, the Plasma administrators will get an alert. The final test against malware is a quarantine machine. Microsoft Azure has the ability to create virtual machines, so the team can upload programs to a preset environment with a few files that should not be touched by the program and then see if the program interacts with the outside files.

Additionally, the team can track internet traffic and see if the program is communicating with someone it should not be talking to.

Plasma may host the program and only do so if they agree. There are even some licenses where asking is superfluous. The MIT license is extremely permissive and the team would be able to take and host software under the MIT if the team wanted without asking the creators. However, to avoid unnecessary complications and to foster a better working relationship with creators, the team will still ask before hosting any software. Additionally, if Plasma only uses a subscription model for making money then the team is not making money from the products. Instead, Plasma is profiting from providing access and marketing for the products.

4.5 Monetization

As mentioned above, hosting all these programs costs money, so Plasma must charge users for the service to continue. There are a variety of ways to monetize Plasma, but some are more viable than others. One of the better ways to make money is to charge users a fee to access the marketplace and auction board, but not the list of previously downloaded programs. Plasma can charge universities and labs an annual fee and then make the whole lab able to use Plasma and could also charge individual users monthly. Additionally, some companies may be willing to offer affiliate links so that Plasma gets a cut of sales found through the marketplace. Finally, Plasma’s marketplace could copy similar services like Steam, which takes a cut of all sales.

Another possible way for Plasma to earn money is to utilize the whitelist. Obviously, selling users’ data would be ethically dubious, but selling access to read the whitelist of acceptable software companies would not have as many uncomfortable questions. For instance, if a recruiter wants a list of verified scientific software developers, they could pay Plasma for a current list. That way they would not have to go through all the researching companies on their own and smaller software companies would get attention. By selling reading access to the whitelist, Plasma would further its primary goal of assisting scientists and scientific research by helping employ more scientific software developers.

4.6 Licensing

Of course, besides security and profit there are other possible concerns to hosting other peoples' software. Some may be concerned with possibly legal issues with using the programs on the store to make a profit. Obviously, the team would not make Plasma if its existence were illegal. To host a program, a team member will ask the creator or creators of the program if Plasma may host the program and only do so if they agree. Even though some licenses do not require permission, to avoid unnecessary complications and to foster a better working relationship with creators, the team will still ask before hosting any software. Additionally, if Plasma only uses a subscription model for making money then the team is not making money from the products. Instead, Plasma is profiting from providing access and marketing for the products.

5 LIMITATIONS

There are some limitations to creating Plasma Software Distribution. From a small team of developers, to other projects in development, to a lack of initial industry experience, and finally to a lack of sufficient documentation for some tools used, Plasma's development had a few hurdles to overcome. First, the small team of developers caused some issues with development speed. With a larger team, Plasma would have been able to grow faster and acquire more features. As for now, Plasma remains a work in progress. Second, there were other projects competing with the development of Plasma due to limited resources, causing blockers and slowdown. Third, Plasma is being created and researched by mostly junior developers, causing hiccups that could be fixed with more experience. Finally, Azure’s documentation on how to integrate its services is not always the clearest, which caused significant delays in certain sections of development.

6 FUTURE WORK

For now, polishing the basic features of Plasma Software Distribution takes priority, but that will not always be the case. Additional abilities and features can help users even further. Such possible future features include a feed for recently published research papers as well as a desktop application instead of running purely on the browser. The research paper feed would require the user to subscribe to a publication and then the user would get notifications as to when new papers have been published. Additionally, they would be able to go to a section on the application and could see the names of the papers. They could then click on the article and read it. As for the desktop application, that would purely be for convenience. The ability to use Plasma outside of the browser would allow for use without an internet connection, which would make launching the user’s programs easier. Of course, those two possibilities are hardly the limit of what Plasma can do, so future market research will be done to ascertain what Plasma’s users want.

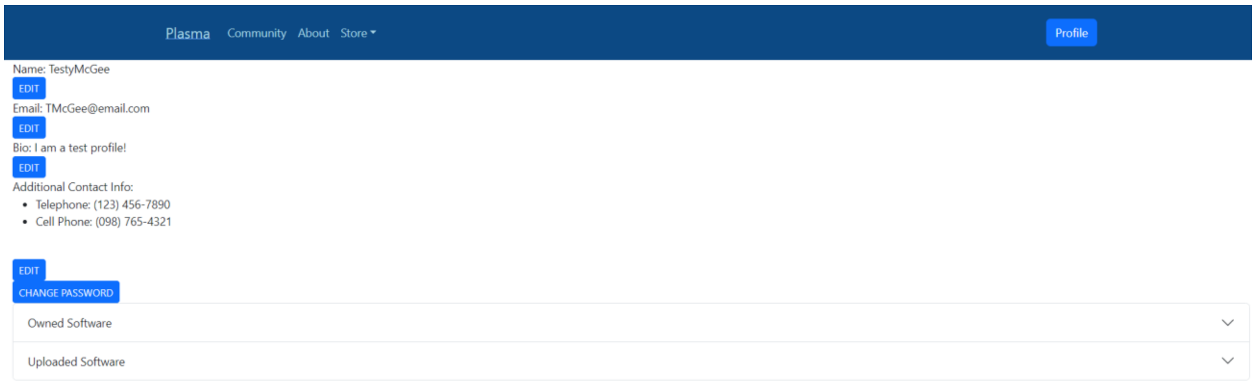
7 RETROSPECTIVE

The creation of Plasma Software Distribution has taught some valuable lessons. Perhaps the most important lesson has been to make more prototypes in the planning phase. Connecting to the Microsoft Azure storage account took a long time, so knowing that it would be that difficult at the start of the project’s development would have been beneficial. Furthermore, such prototypes would have allowed for more practice with JavaScript which would help with some of the difficulties understanding how the language operates. Finally, the project was rather ambitious. Starting with goals that were easier to achieve and deliver within the required time and improving the project once those milestones were accomplished would have led to a smoother start.

Although these lessons learned from Plasma's production cannot help the initial production, they can help the future of the project. Plasma will continue to grow and improve so that its users can have the most fitting tools for their research. By utilizing lessons learned from previous literature and conducting its own research, Plasma has worked to improve its service to others as well as its ability to sustain itself. Furthermore, by using third-party APIs and Microsoft Azure, Plasma can focus on creating what matters instead of reinventing the wheel. Although other software distribution or freelance job services exist, few fall into the same niche that Plasma will fill.

By using Plasma Software Distribution, researchers, scientists, and professors can more easily get the tools they need. Not only that, but freelance programmers can make money and contribute to scientific innovation. Labs, universities, and solo researchers will find that Plasma is a significant boon to their abilities. From Plasma's research, to creation, to service of others, the goal of Plasma Software Distribution is to be linked with and advance scientific progress.

8 APPENDIX



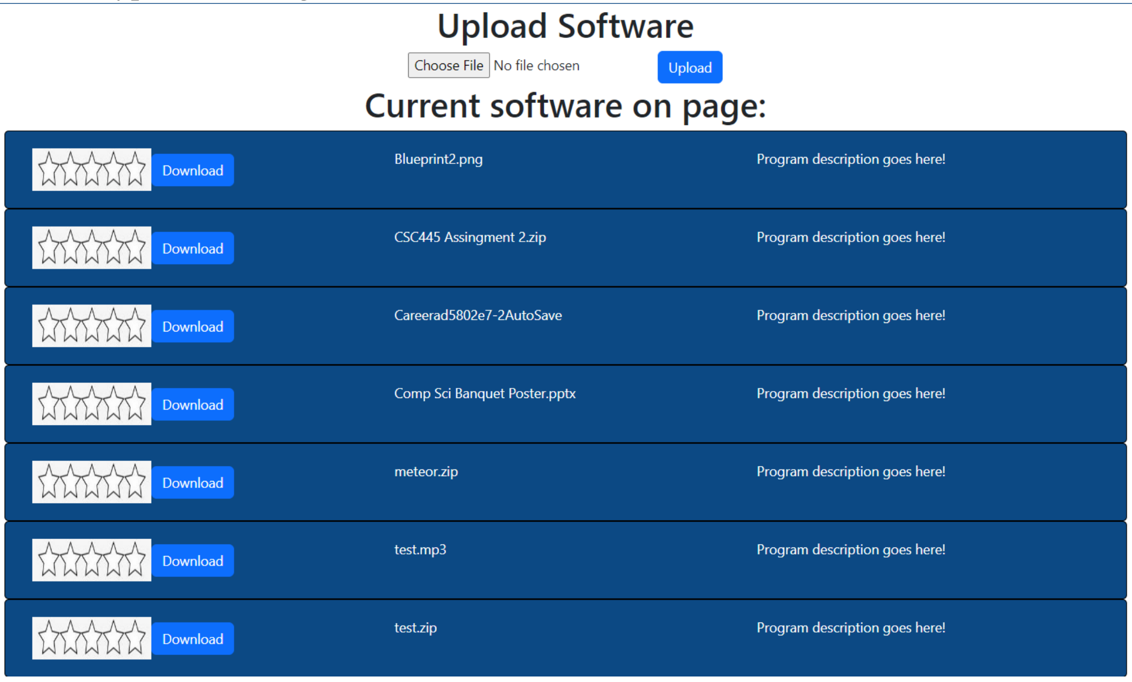
Name: TestyMcGee
EDIT
Email: TMcGee@email.com
EDIT
Bio: I am a test profile!
EDIT
Additional Contact Info:
- Telephone: (123) 456-7890
- Cell Phone: (098) 765-4321

EDIT
CHANGE PASSWORD

Owned Software

Uploaded Software

Figure 1: Prototype Profile Page



Upload Software

Choose File No file chosen Upload

Current software on page:

Download	Blueprint2.png	Program description goes here!
Download	CSC445 Assingment 2.zip	Program description goes here!
Download	Careerad5802e7-2AutoSave	Program description goes here!
Download	Comp Sci Banquet Poster.pptx	Program description goes here!
Download	meteor.zip	Program description goes here!
Download	test.mp3	Program description goes here!
Download	test.zip	Program description goes here!

Figure 2: Prototype Software Upload Page

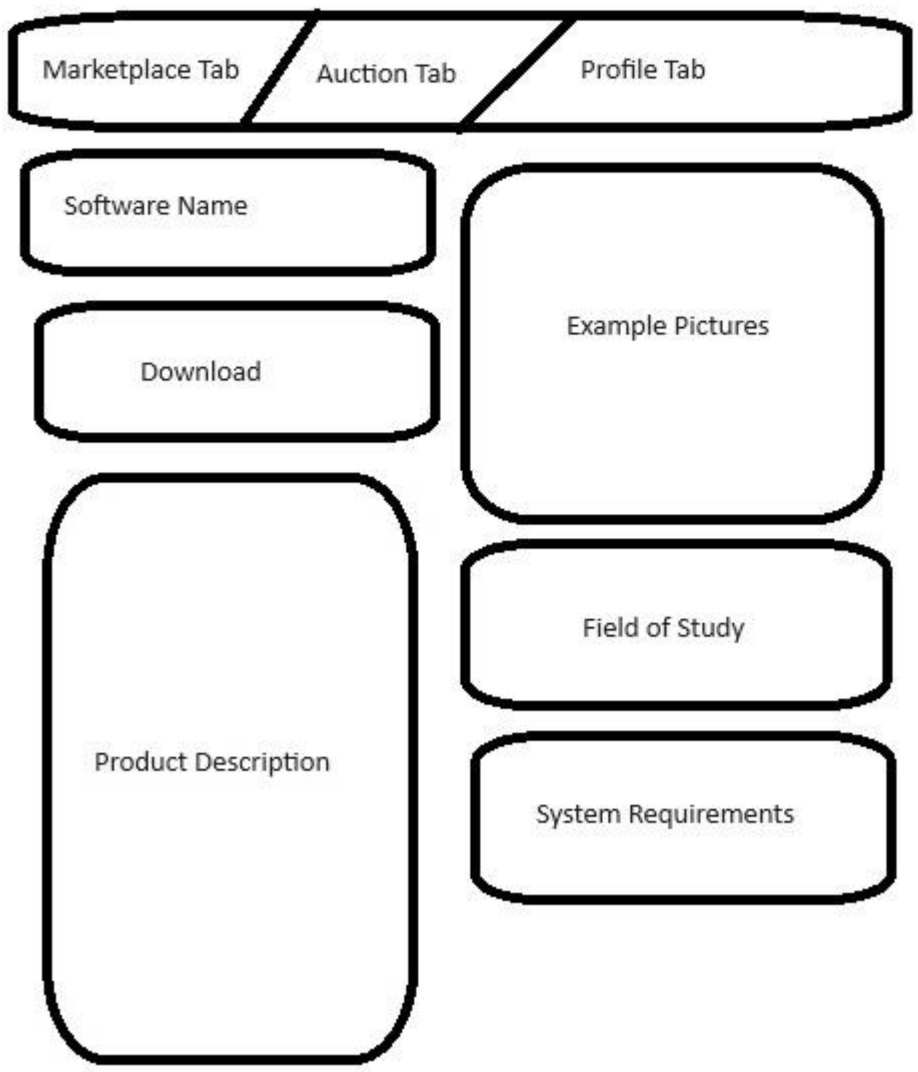


Figure 3: Software Download Page

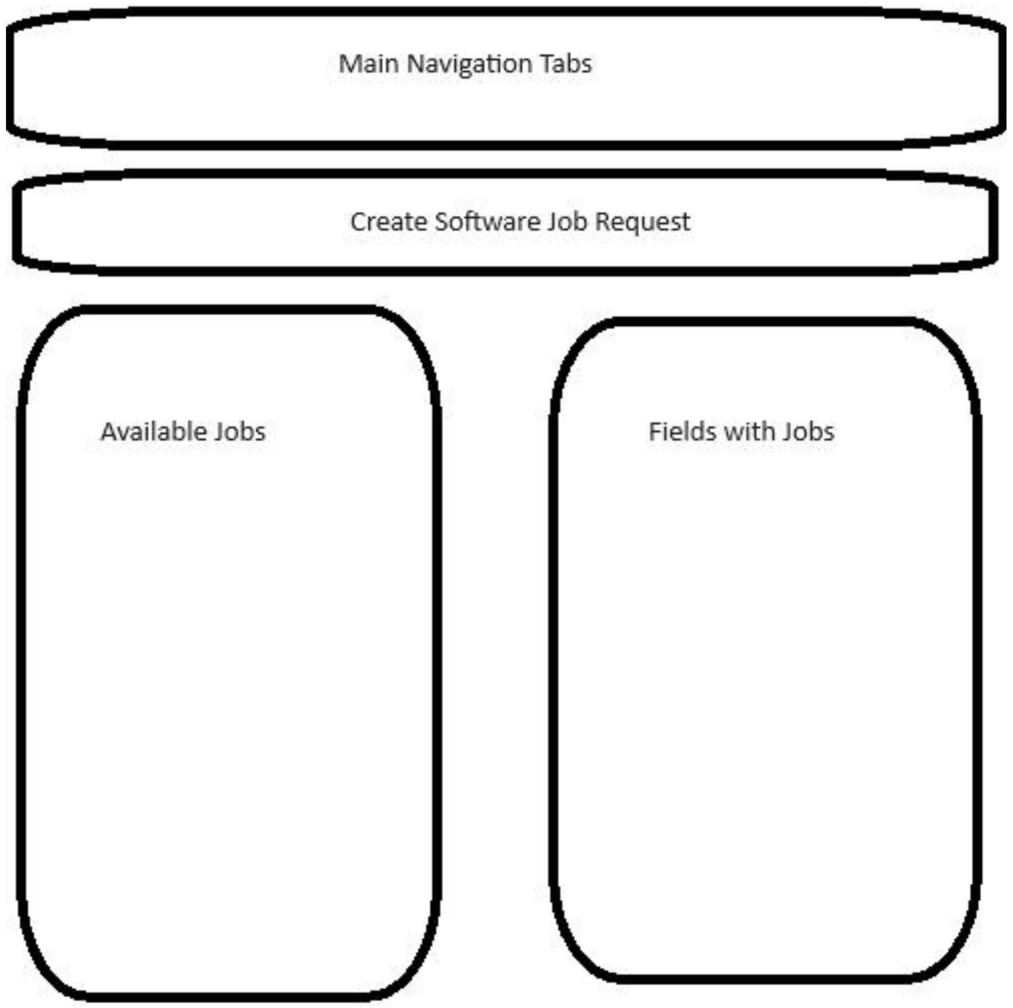


Figure 4: Job Auction Board Mockup

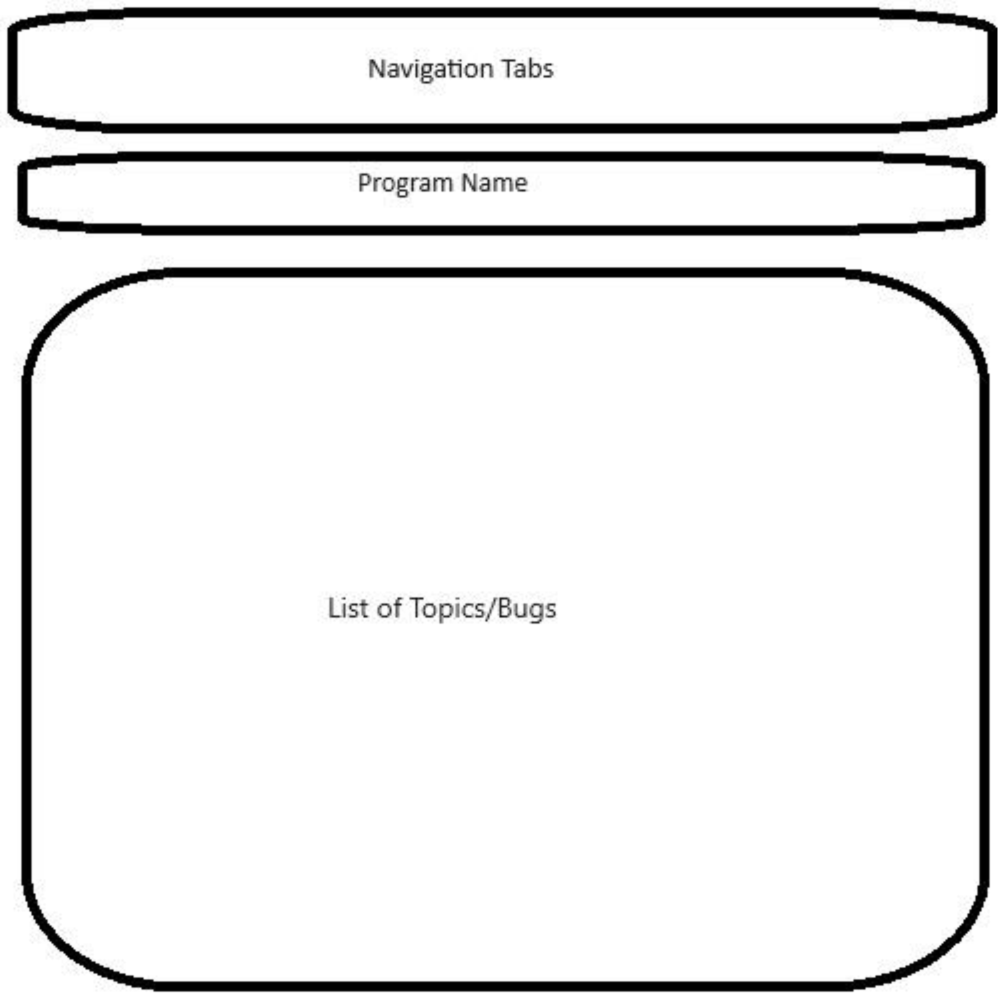


Figure 5: Product Discussion and Bug Report Page

REFERENCES

Belgin M, Perini TA, Liu F(C), et al. A data-driven support strategy for a sustainable research software repository. *Concurrency Computat Pract Exper*. 2019; 31:e5338. https://doi-org.ezproxy.waterfield.murraystate.edu/10.1002/cpe.5338

Bernstein, Philip A., and Umeshwar Dayal. "An overview of repository technology." *VLDB*. Vol. 94. 1994.

Shukla, S., Bisht, K., Tiwari, K., Bashir, S. (2023). Data Monetization. In: Data Economy in the Digital Age. Data-Intensive Research. Springer, Singapore. https://doi.org/10.1007/978-981-99-7677-5_3

A. E. Hassan, "The road ahead for Mining Software Repositories," 2008 Frontiers of Software Maintenance, Beijing, China, 2008, pp. 48-57, doi: 10.1109/FOSM.2008.4659248.

Ö. A. Aslan and R. Samet, "A Comprehensive Review on Malware Detection Approaches," in IEEE Access, vol. 8, pp. 6249-6271, 2020, doi: 10.1109/ACCESS.2019.2963724. keywords: {Computer viruses;Feature extraction;Encryption;Internet;Cyber security;malware classification;malware detection approaches;malware features},

P. Yang, N. Xiong and J. Ren, "Data Security and Privacy Protection for Cloud Storage: A Survey," in IEEE Access, vol. 8, pp. 131723-131740, 2020, doi: 10.1109/ACCESS.2020.3009876. keywords: {Cloud computing;Encryption;Data privacy;Secure storage;Memory;Cloud storage;data security;cryptography;access control;privacy protection},

Moore, Connor, and Wesley Totten. "Scientific Software Use." 13 Feb. 2024.

PACE. Partnership for an Advanced Computing Environment (PACE); 2017.

"Steam Store." *Steam*, 12 Sept. 2003, store.steampowered.com/.

"Fiverr - Freelance Services Marketplace." *Fiverr*, 2010, www.fiverr.com/.

"OSSR." *ESCAPE*, CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE CNRS, 1

Feb. 2019, projectescape.eu/ossr.