CSC400

Heroesdonate.site

<https://github.com/Jason-Ponce/HeroesDonatePackage>

May 17, 2019

Jason Ponce

Poncej2@southernct.edu



**HeroesDonate**

**By Jason Ponce**

# Introduction

For my CSC 400 project. I choose to fix a problem I’ve faced before in a creative way that would not only help others in my situation but make a difference. I’ve been gaming on my PC for the last eight years and over those years I’ve accumulated copies of video games. It’s simple enough to gift these copies to friends. Except, there has been times when my friends already own the games I want to gift. So these games stay in my library doing nothing. I could setup a giveaway or make it into a puzzle for the first person to crack and get the game. I would like to give the chance for others in my situation, with any sort of gift able digital good, be able to give away their items in a website. This website, my project, I call HeroesDonate.

HeroesDonate is a digital good donation website. My inspirations were Goodwill and DonorsChoose.org and my website HeroesDonate is a mashup of both inspirations. I would like HeroesDonate to be the website individuals with excess digital goods go to. To donate and make sure their digital goods go to a good cause. It can be as simple as going to the website then submitting their code and watch as their digital good gets donated. The two main users of the website will be Donors and Organizations. With a third supporting role of the Admin to help facilitated and ensure that there are no problems. Donors are users who come in with their digital goods and donate. Organizations are charities that have a want for certain digital goods. They can post a fulfillment of digital goods that Donors can see and directly help with achieving in those fulfillments.

To put simply these are the core features.

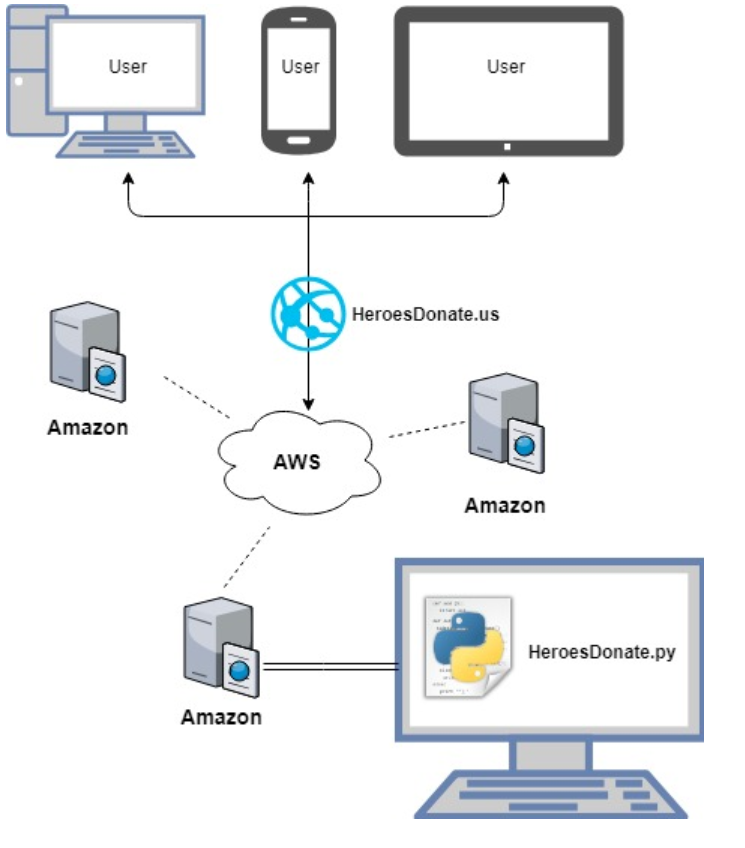
* Donors can donate digital goods, with confirmation they made a difference.
* Organizations can request certain digital goods to be fulfilled based on their needs.
* Admins can link up anonymous donations to current requests.
* A ledger to publically show that goods donated are being sent to the appropriate user.

One of the core features listed, the ledger, is a necessary requirement for users to verify that their digital good is being used in their intended way. Whenever a digital good is submitted, the characters are encrypted for security and to ensure that the server owner/admin will not be able to use the digital good. This encryption will be able to be seen publically in the ledger. For the initial scope of my project, I would like to make the ledger functional in any way possible. With the intention to implement block chaining to fully secure the ledger. This way, each user is held accountable for the ledger and accessible by any of the users. Allowing for a fully transparent but encrypted p2p network for the digital good.

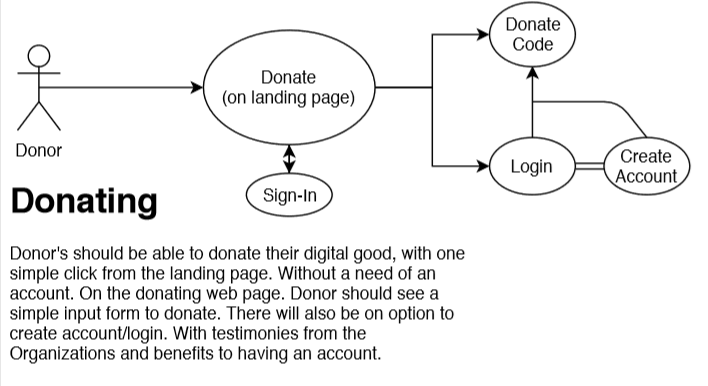
# Architecture

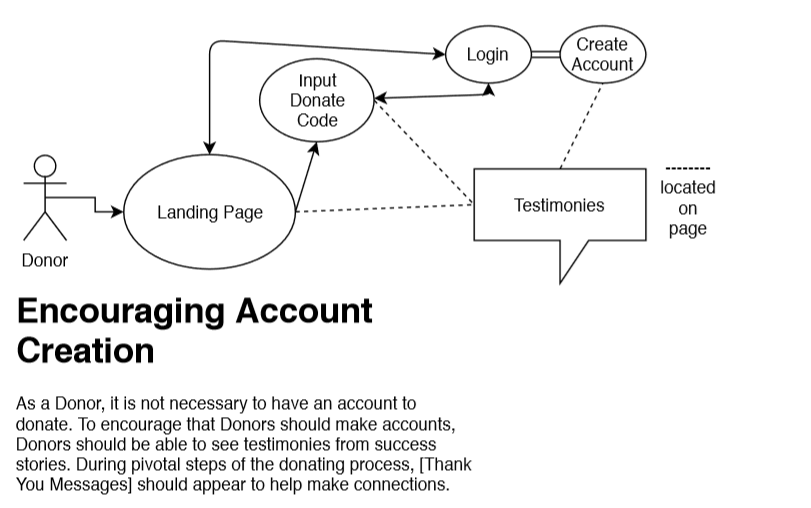
HeroesDonate will run on a client server side system. Webhosting will be done by Amazon AWS as it’s one of the largest cloud based web hosting companies in the world. Using AWS, the web app will be running 24/7 unless AWS itself experiences a problem. Where in the situation if HeroesDonate is hosted on my home network. Various problems can crop up and shut down the web app. This is important because the server for HeroesDonate will rely on a 24/7 uptime to process information such as receiving new donations, storing new donation requests and containing user information. Client side, each user will be asking for donations, page, information requests that will need to be able to have access to the server at all times.

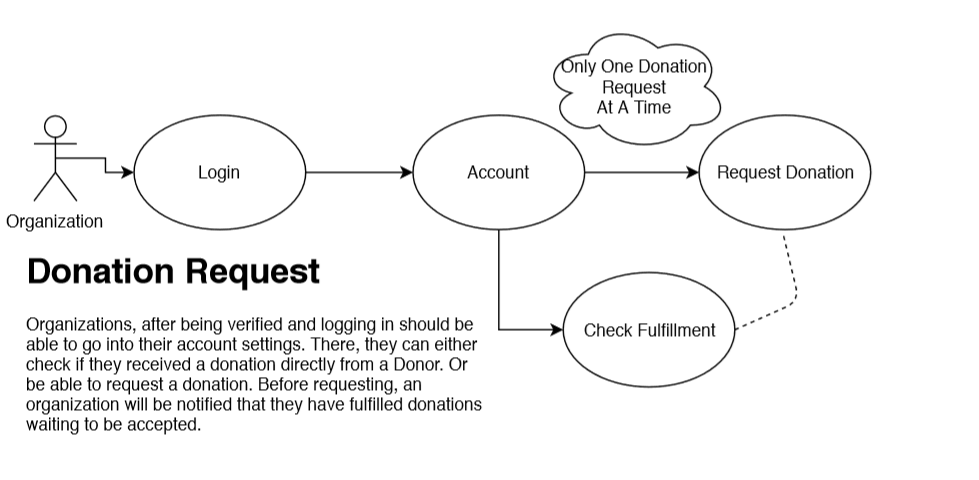
HeroesDonate will use Python as its main programming language for its backend for its ease of use and its wide support for setting up a web app. An important module to be used will be Flask. Flask is important for its micro web framework and ease of expansion. For the front end of the web app, MaterializeCSS a framework based on Google’s Material Design, will be used. Bootstrap was also under consideration, but MaterializeCSS exemplifies its elements in a way that users with no familiarity with the website could pick up on the cues on the UI and understand how they work. HeroesDonate is intended to be used on both mobile and desktop interfaces. A seamless experience is necessary to help facilitate the feeling that from the comforts of your phone or at the desk with your PC, you can donate.

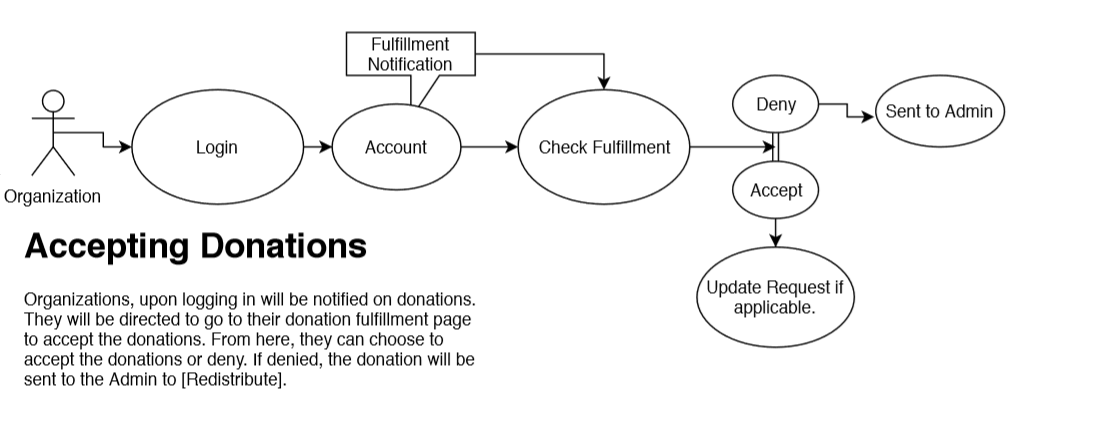


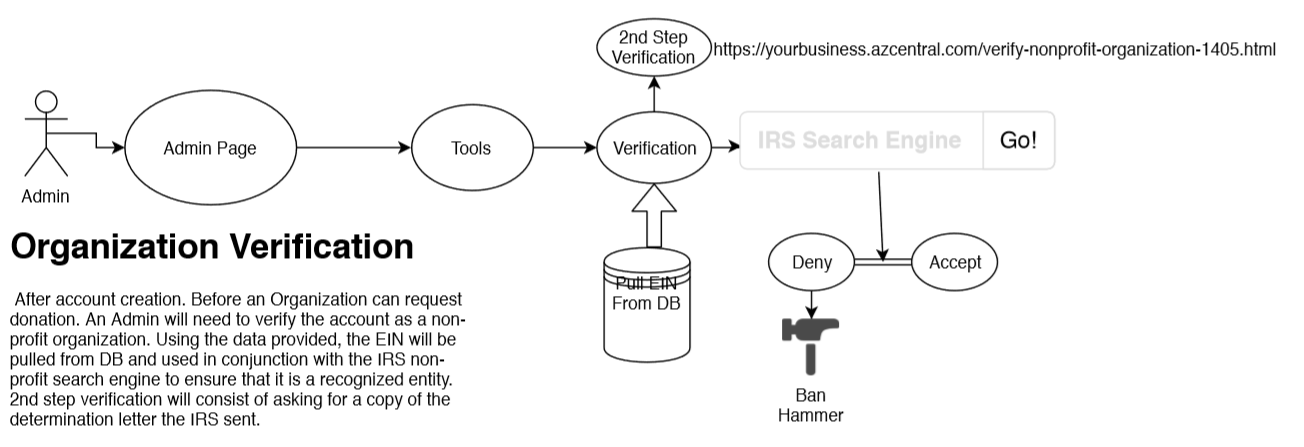
# Use Cases

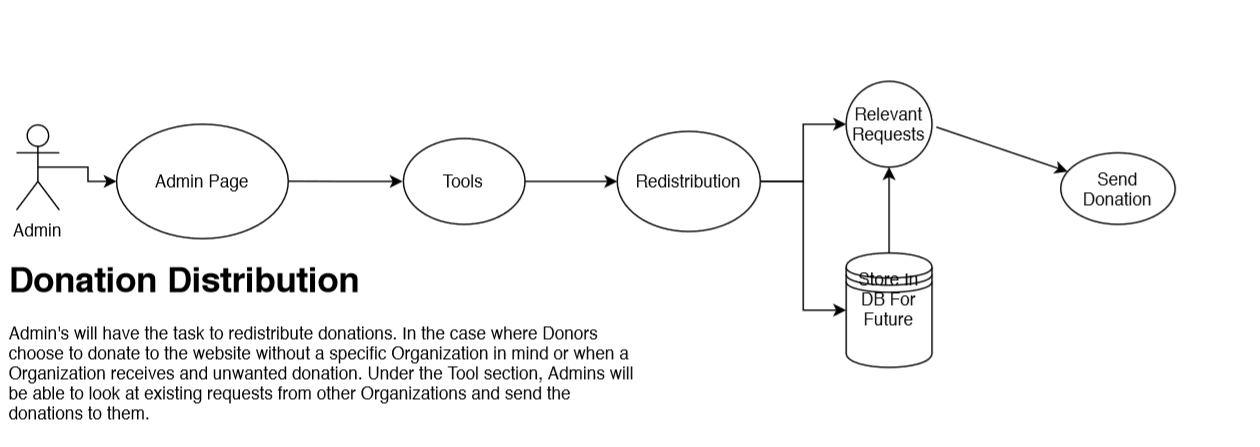




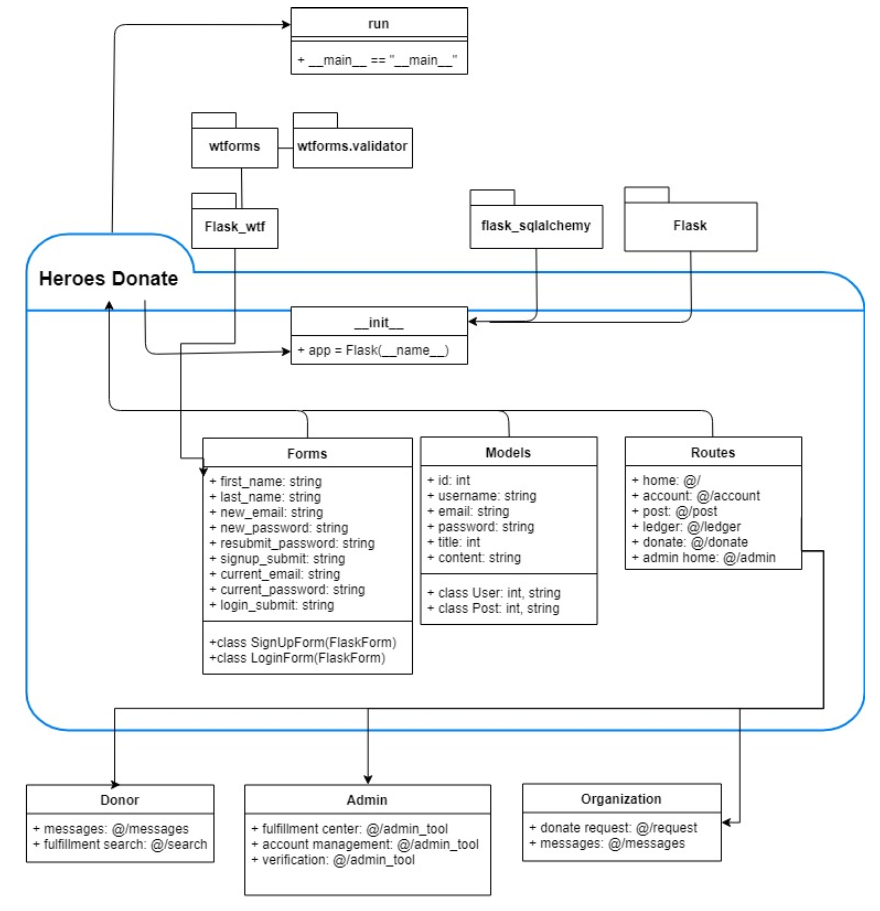




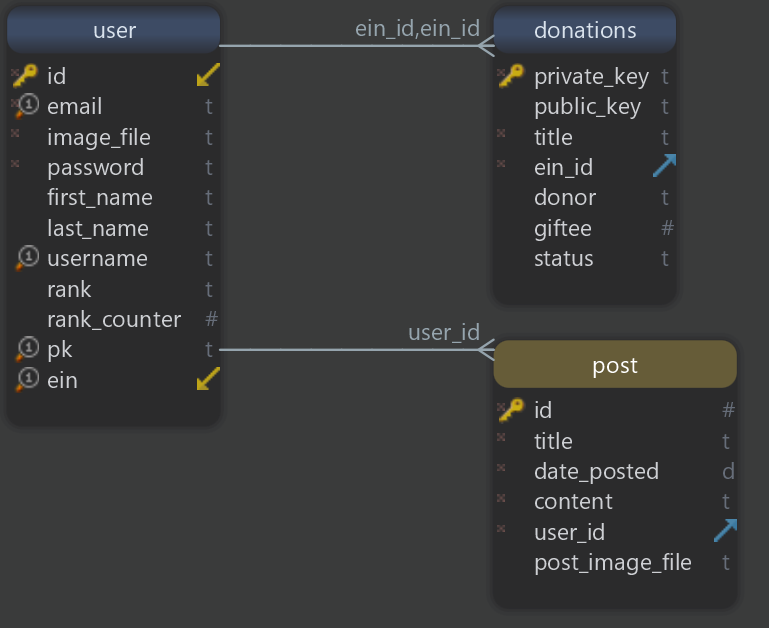




# Structural Design



# Data

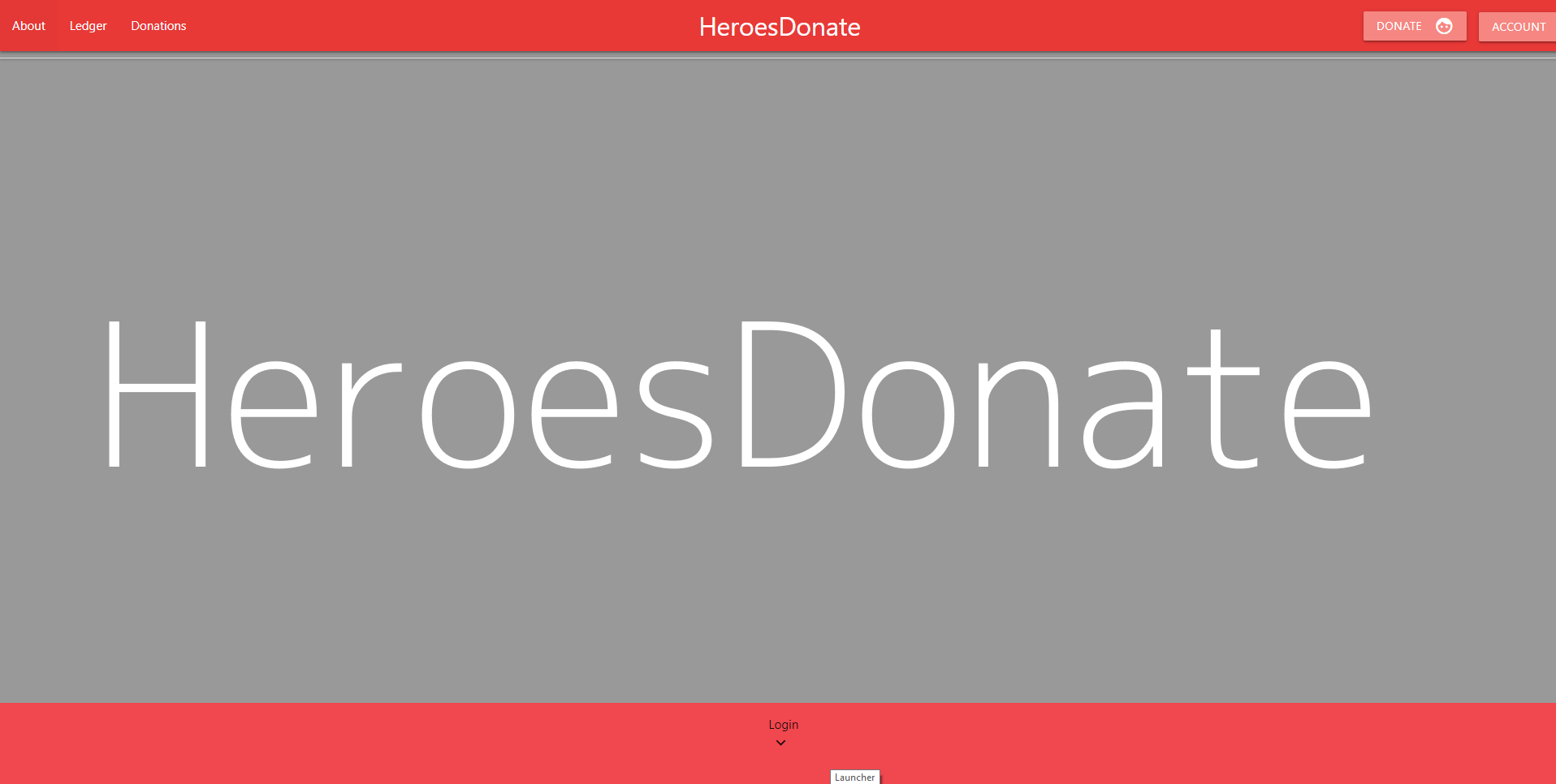


SQLite Schema

# User Interface Design

With my project, I decided to use MaterializeCSS “a modern responsive front-end framework based on Material Design”. One of the reasons why I choose MaterializeCSS over other frameworks. Is that the framework provided gives a more physical feeling to its components. Based on Google’s design philosophy on its web work. Is that each digital material that is created should feel as if it’s an object that you can touch and intractable. Such as with any object in the real world. Another benefit is the uniformity and ease when transitioning between different devices.

The following UI screenshots are the current progress UI of the project.



Landing Page

**Landing Page**

This is the current look of the website with the colors being placeholders. After adequate work on the backend is finished. The website will be redone to better align with a heroic look. Following along with a motif that is explained better here <http://comicsalliance.com/superhero-color-theory-primary-heroes/>. More easily explained, HeroesDonate will use bright primary colors and be more akin to a comic book. HeroesDonate will also utilize the framework’s strength in mobile use. When the browser detects a screen below 768x 882, the links are changed into a hamburger icon. Once clicked the side nav-bar becomes available for mobile use.



Side Nav-Bar

**Navbar**

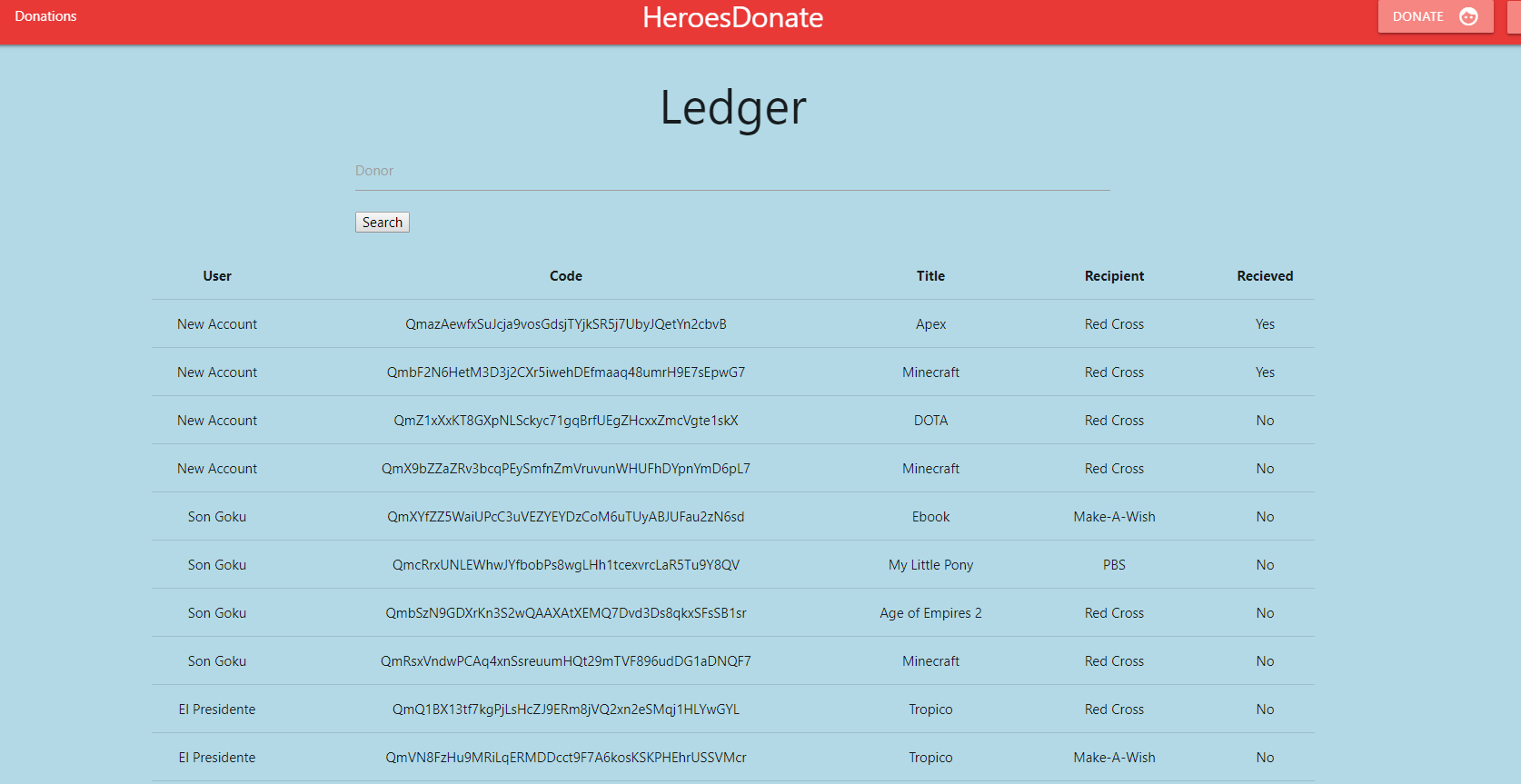
Inside the side navbar, the previous links showcased in the landing page will be shown here instead. There will also be a location for the account profile of the user to be shown up here. If not logged in. This will lead the user to login/create an account. Below in the login/form picture, is the general layout of what a form will look like. In the case of the login/account creation, both sign up and login are available to the user without having to be directed to another page. To all users, this form is available with the account button in the navbar of the landing page.



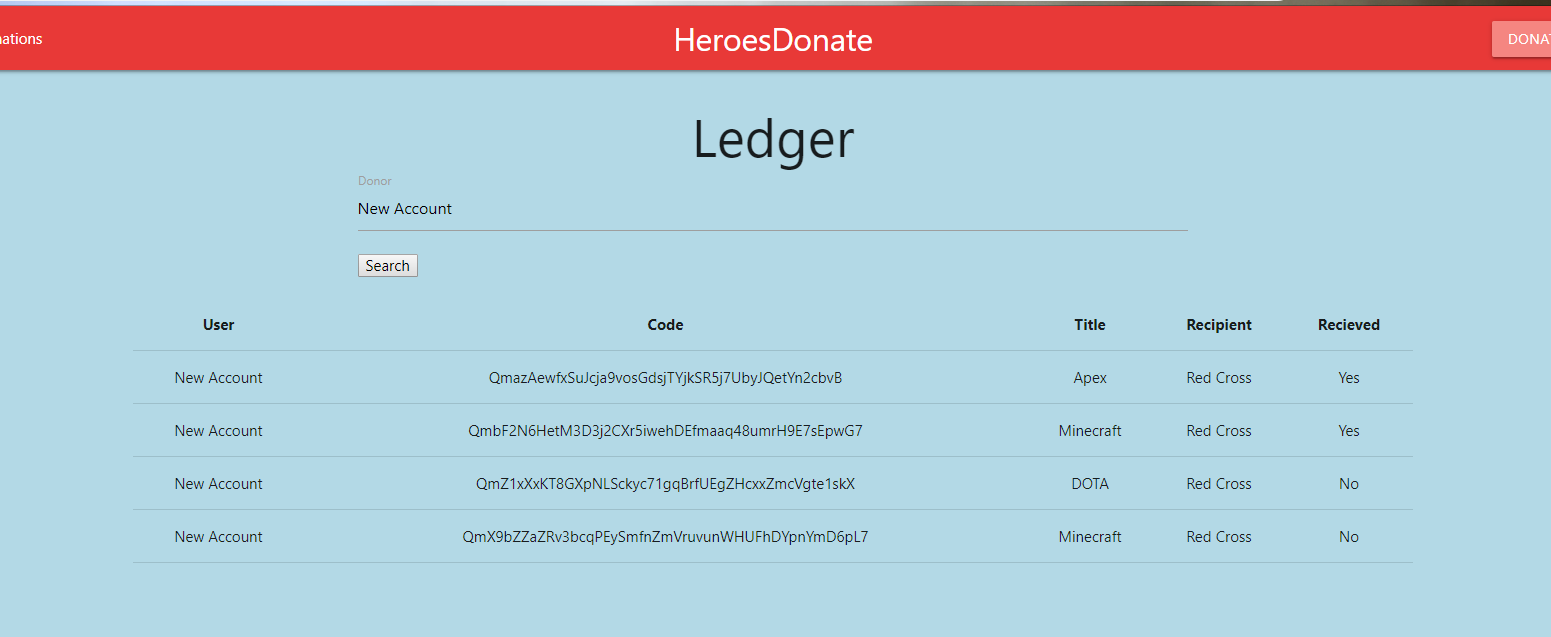
Login/Forms

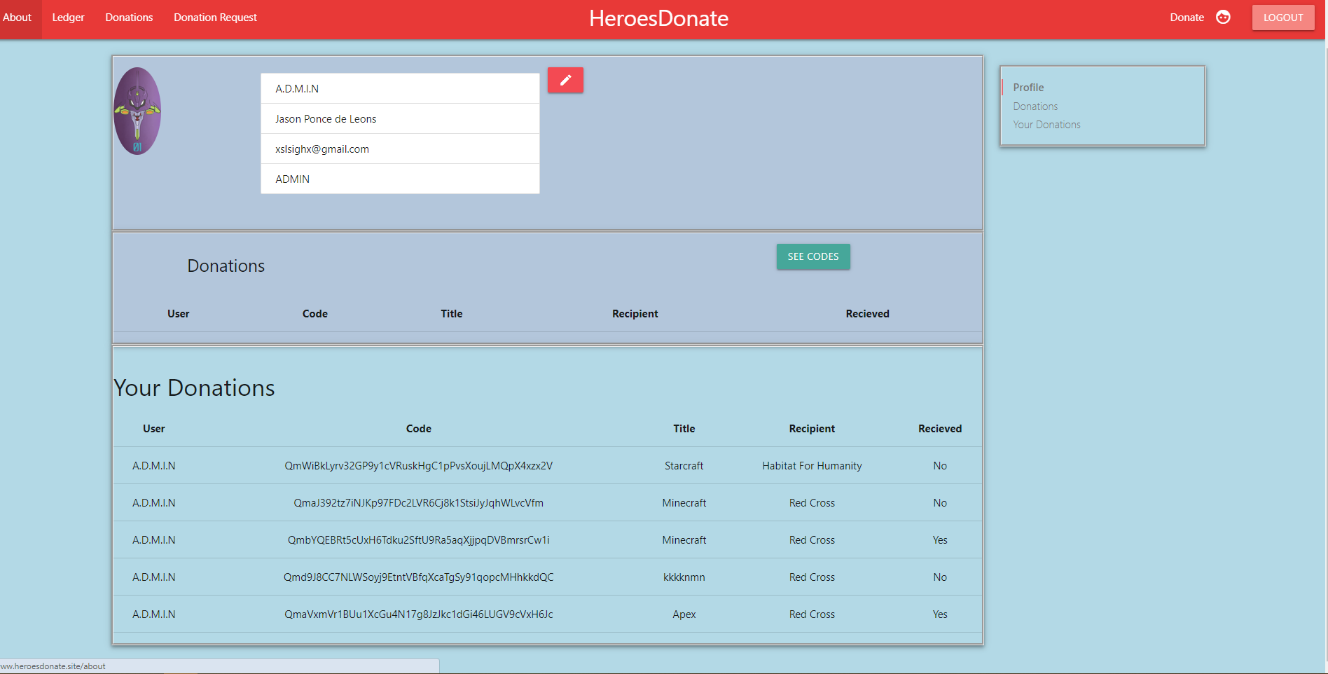
**Ledger**

The ledger screen shots were also included to showcase the search functionality of donations. Whether signed in or not. Any user can track the progress of a donation. Once the organization, i.e. the recipient, accepted the donation. The status is changed from No to Yes. This is also reflected in the Account page of each user involved with each step.

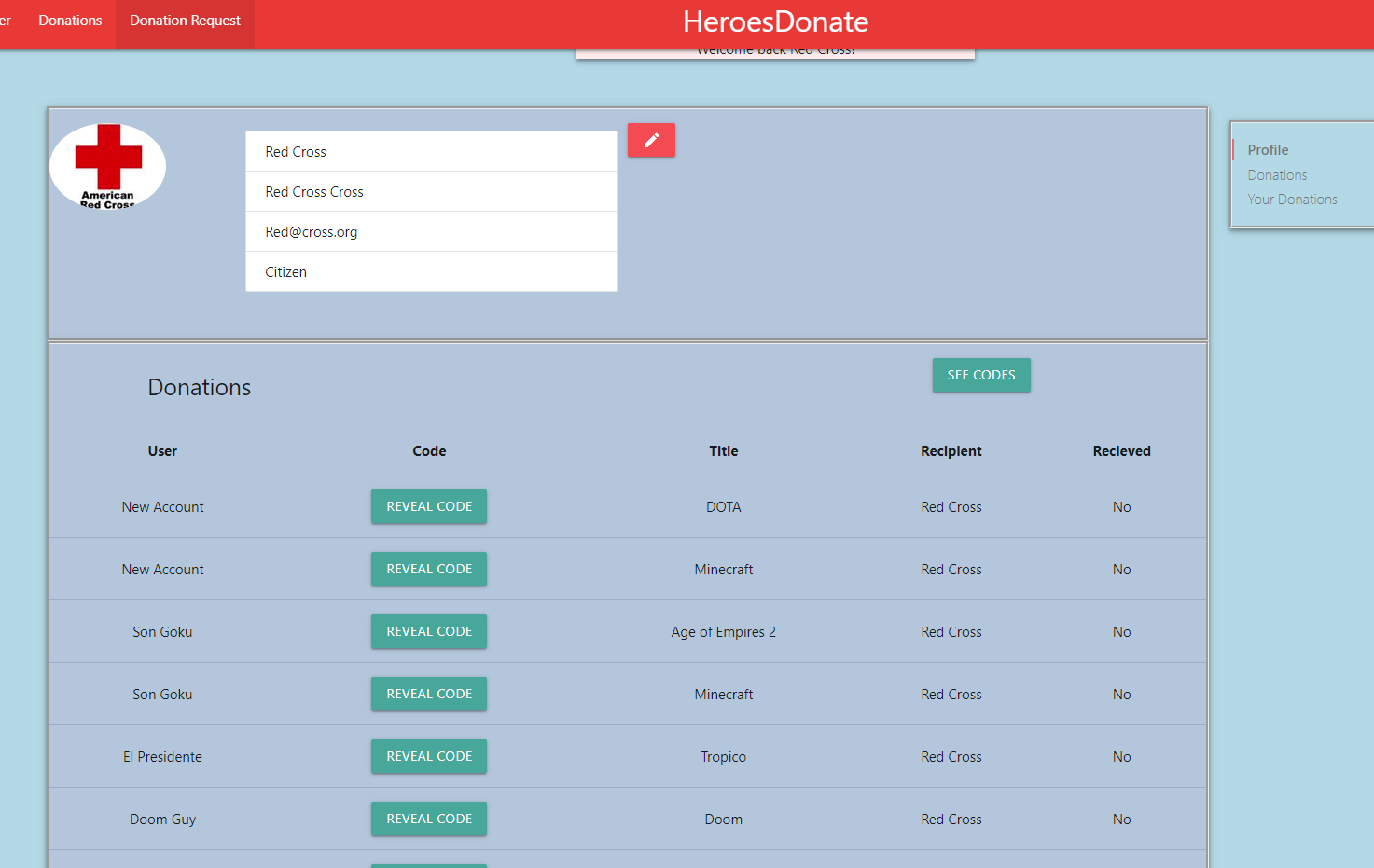


Ledger Without Searching



Ledger With User Search

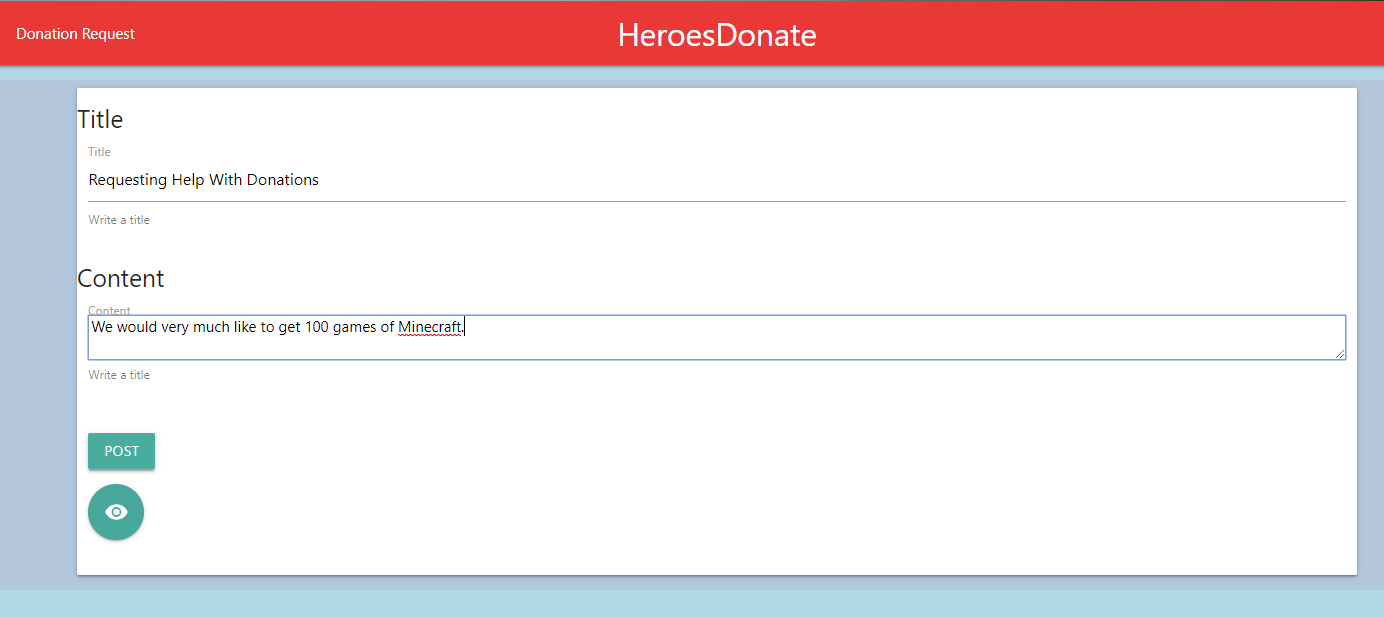
Account Page Donor



Account Page Organization

**Account Page**

When a donor logins in. They are able to see the status of their donations instead of having to search the ledger. Likewise, when an organization logs in, they are able to see the inverse. Donations that came in are shown. The difference is that the code is “hidden” underneath a Reveal Code button. Once the button is pressed, the hashed link containing the code decrypted and is sent to the organization requesting. At this point the data is saved on the local database for the organization to use at their leisure. The organization can now see all their codes whenever they please with the See Codes button.



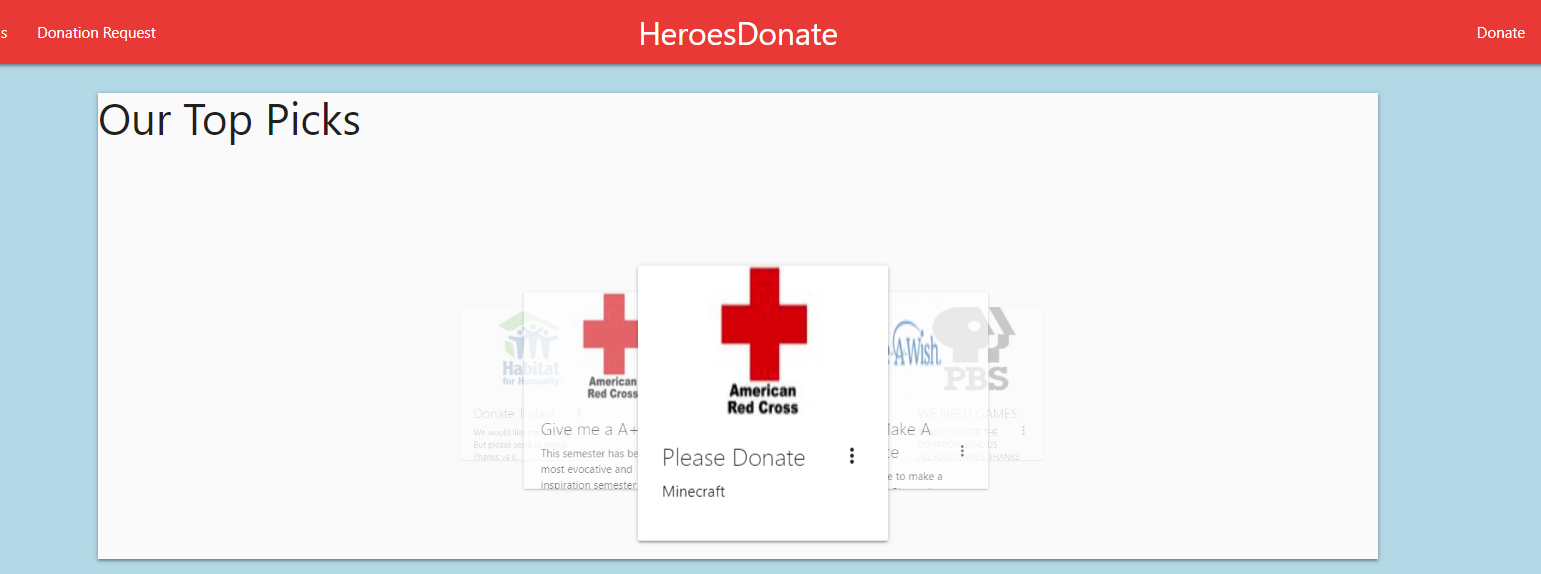
Donation Request



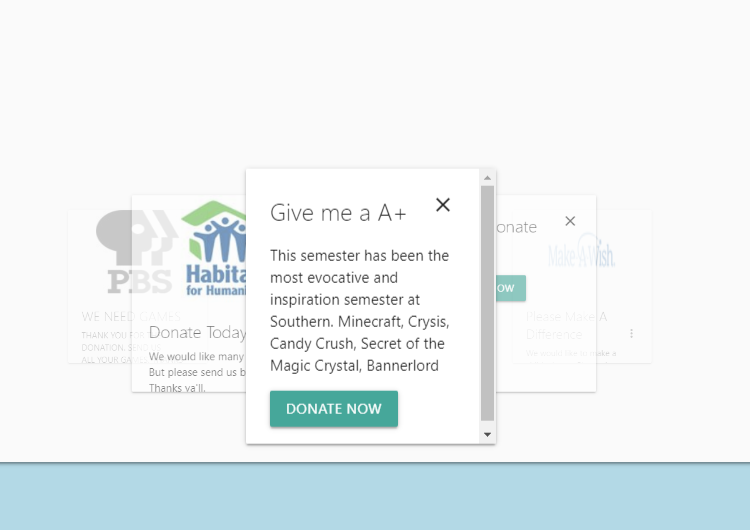
Preview

**Donation Request**

Once logged in, an organization can make a post to request a donation from donors. They simply create a title for the request. Next they add the content of the request. Such as the reason they would like the donation and also what specifically they would like to receive. An organization could also preview how their post will look like with the button with the eyeball icon. Using Javascript the content of each of the forms is saved to the session storage of the browser. This is a temporary storage that will be deleted once the tab or the browser is closed. At the launch of the modal in the preview button, the session storage data is added to the contents of the card that is used for the Donations. The image used for the donation request is added automatically based on the profile image the user has at the time of submission.



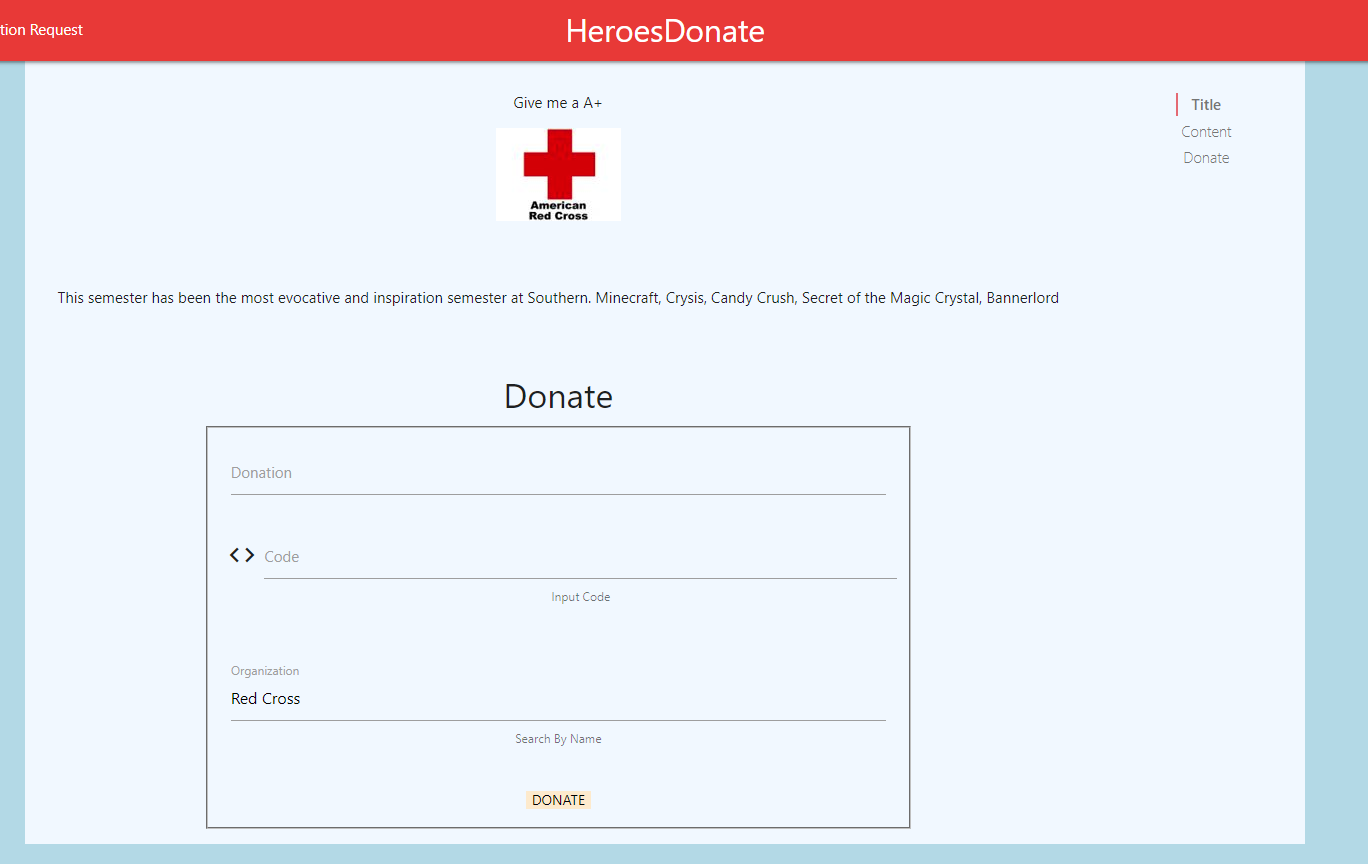
Donations



Clicking on a Donation

**Donations**

In the donations page each of the requests is added into a carousal. The idea was that there was going to be multiple carousal for different categories such as top picks, important request, ect. When a user clicks on the donations request, the request pops with more information and a donate now button. For future versions of this project. I would like to expand this page so that the request for donations is separate from the content of the post. In the Donations page of the project, each donation should be dynamic and be able to update the request once a donation is confirmed. Currently each donation request is static and does not reflect changes.



Donate Now

**Donate Now**

After a user clicks on the “Donate Now” button, they get sent to this page, some of the info on this page is prefilled with data. The Organization data is hardcoded in the backend so if the user tries to change it for some reason then it won’t be saved. From here the User can type in the name of the donation, the actual code for the donation and click submit! From each page the User can also donate but the difference between the pages of Donate and Donate Now is the prefilled data.

# Detailed Workplan

# Lessons Learned

I do not intend to bring this project to fruition. One of the biggest problems that I had thought of when first submitting the project deals with verification. A user can donate a digital code in the format of “XXXX-XXXX-XXXX-XXXX”. My options would be the following.

* My options to deal with this verification would be either to accept the donation myself and claim it. At this point, I would own the game and transfer would not be possible.
* My next option which is limited. Is to use the creators of the digital donation to verify the donation itself. Microsoft allows this in their own website: [Microsoft.com/redeem](microsoft.com/redeem). This would mean I would have to check each code as they are submitted with either site tools I built or a script to automate the task. Easily Microsoft could see this as a DoS attack and block me. Also not every creator has a redemption checker, like Steam.
* The third option is to build an API for code redemption checker that creators can go ahead and be a part of. This is the most unfeasible option.
* Lastly, I can take each donation as is. Once an organization accepts the donation and reports back that it’s a working donation. I can track users and build a “confidence” level to predict the chances of successfulness. This leads to another problem.

After an organization accepts a donation. They can easily say that the code doesn’t work. Even if it does work. Either maliciously or due to lack of technical ability. A solution to this would be a user risk warning, essentially they get what they get. This would not build confidence in the website and might be too much of a risk to early reputation.

Another major problem with donating is the scale of donations we currently accept. There is no set standard on how a code is given out from a creator’s website. I envisioned a website where you go to the landing page. Press a button, input your code and send. But to facilitate each donation, multiple checks and filters would have to be placed in a single form. Or there would have to be multiple forms to take the user to the correct one that accepts the format of the code they’re donating. If there are no checks and filters, then the website is a very elaborate, contrived, messaging system. In the donation form, as a donor I could write “Please wake up sleeper agent 47” to *Shady Shell Organization* and the organization can accept the donation and read the message. Understandably in this case it seems farfetched. But I’m potentially opening myself up to unwanted legal consequences.

If I had to do this project again, management of my personal time and school work would need to be worked on. I started working on this project after our first CSC400 class. I was worried about the status of the CSC330 project and I needed as much time as I could possibly get. During the month of March I reached a point where I believed my project was way ahead of schedule. At this point, work started slowing down and with projects and exams the progress of the project slowed down to a crawl. Given another semester with nothing but CSC400. This project would have been finished with all goals in mind.

# Future Version

I will be working on this project on my own time. I fully intended to get a working website with the functionality I had originally promised at the beginning of the semester. Various things that need to be worked on.

* Separate the different users into three, Admin, Donor, and Organization. Currently any registered user can perform and operate any of the services that were designed for each other. This would be a short term goal that I would like to reach soon.
* After different rolls are created, I will need to restrict the webpages based on the needs of each user. For example a donor should not have access to the donation request system. As well as this goal I would like to finish soon.
* The admin users need their tools. When an organization registers, there is no verification process to make sure that they are who they say they are. Admins should be able to retrieve the data from the IRS database with the information given. Along with requesting a copy of their determination letter.
* Getting a local blockchain running in conjunction with Solidity and Flask.
* Upgrading the look of the website with better CSS and Javascript and following through with the original hero theme.

Ultimately all of these features would be the long term goals of the project. I would like to present HeroesDonate.site to prospective employers through my personal website. Even if I manage to hit all my short term goals. Refactoring of the code will be needed and in general, practicing new features with the website will be done.

# Lifelong Goals

Prior to taking CSC 330 Software Design and Development during fall 2018, my outlook on programming was terrible. I hated programming and through the years leading up to CSC330, I let my skills in programming slack to the point that I could barely remember the basics. Leading up to CSC330 I decided to change that. I stepped up and lead my team to creating and just barely pushing a “completed” project. I learned to write backend with Flask, I picked up HTML which I had never written before, and I learned how to put it all together in a team. But the following semester I had signed up for the CSC400 Capstone class. With the progress I had made in CSC330, I knew my skills wouldn’t be enough to do what I previously accomplished with a 5 man team by myself.

Because of my time during CSC330 I learned I wanted to be a front end developer with the ability to do the job of a full stack developer. Originally I wanted to be a System Administrator. With the cloud becoming more enticing. Less and less companies are needing SysAdmin. By becoming a front end developer I can work on the creative drive inside of me and have a good paying job. To meet this future I will need to work more on my skills.

With winter break leading up to CSC400. I picked up classes on Udemy and started to learn how to properly manipulate the DOM with Javascript. I focused more on Flask and learned some tricks in CSS. Now at the time of this writing, with my capstone project coming to an end. I’m looking towards my future as a CSC student and knowing how my limitations are stopping me from becoming a better developer.

I’m extending my stay at Southern Connecticut State University for an extra semester for two reasons. During fall 2019 I will be obtaining a minor in studio art. The minor will be focused on the graphical arts. With the help of Professor Girard of the Art Department. He helped me choose classes that would help me to learn more about the UX design of websites and Professor Girard <https://www.alexjgirard.com/resume>, is accomplished with those skills. During this period of time I will be practicing more of my skills on web design without the constraints of a project as well.

The skills I learned during CSC330 and 400 might be just barely enough to get a job. But I don’t want to just get a job barely scraping by. I want to go into an interview confident enough to be proud of showing my work. I want to be able to show future employers that the skills I learned was because of SCSU. I have plenty of catching up as a 26 year old and I need to be better than younger applicants at a job. With time this will get even more difficult. So I don’t see CSC400 as just another “project” that I need to get done so I can get my diploma. I see it as pivotal moment in my life to show me that I can achieve so much more if I apply myself and become a better developer.