Capstone Proposal

Jordon Bowden

Neumont College of Computer Science

Table of Contents

Summary…………………………………………………………………..………………………3

Introduction………………………………………………………………………………………..5

Program and Tasks

* Programs…………………………………………………………………………………10
* Tasks……………………………………………………………………………………..11

Budget……………………………………………………………………………………………12

Task Schedule……………………………………………………………………………………13

Experience………………………………………………………………………………………..16

Works Cited……………………………………………………………………………………...18

SUMMARY

As the leader of the transcendentalist movement, Ralph Waldo Emerson once said, “To be yourself in a world that is constantly trying to make you something else is the greatest accomplishment.” Society’s attempt to change us is an everyday struggle we face in the more developed parts of the world, as Emerson sees it. Society can look at something and decide that it is not what is commonly accepted, defining what is “normal”, further explained by Kendra Cherry (2018). This persists to the online world, to an extent.

In some places, you are limited to who you can be, whether that be restricted words, limiting features, or someone taking your spot. What’s even worse, is when that person who has taken your spot, leaves, but still occupies the space with memories of themselves. This is referring to the fact that users can take whatever name they desire if they come first, never to be claimed again by anyone else, even when they stop using the service. Instead of this going unnoticed, we can remind those of the accounts they have given up, giving them a chance to restore themselves or relinquish their identity.

Notifications can be a quick, easy reminder, and not persistent enough to be seen as intrusive. Knowing all of this, I plan to make a system that monitors all user activity on the services that I have paired with, and offers to sell the username that has been inactive on this new bidding website, similar to how people of Atlassian allow notifications to be sent after a user goes inactive. It will notify any inactive user of the supported services that people want their name, and give them an opportunity to give it to the buyers for compensation. This will be seen through because all the knowledge needed has already been gathered. A board of user information will need to be created, full of hundreds, even thousands of usernames that have been inactive, allowing them to be sold. If there are things that still need to be learned, however, knowledge could easily build off the strong foundation built here at Neumont. This cannot be done without the knowledge and consent of superiors. I ask you, my chair head, to understand this plan and allow the progression of it to continue, eliminating the pseudo-identity theft running rampant on the internet, and allow Sinomim to be produced.

INTRODUCTION

This may be looked over by many people, seen as such a small and menial problem for only people that use websites. 54.4% of all the world’s population has access to the internet (Miniwatts Marketing Group, 2018). Knowing this, we can only assume that the internet has been used by everyone with internet access, since there is no way of confirming that it has been used previously. Those people can browse, according to their country’s rules and regulations pertaining to usage of internet. Although there are certain sites that cannot be accessed everywhere, there are always things that are available for them to do. With most sites with interactivity and information sharing, there are accounts to show who is doing what and what they are allowed to do. On some of those sites with accounts, duplication of certain fields is not available, such as usernames.

Just to be safe, we will only use about half of the population that has internet access (25% of world population) based on the information Miniwatts Marketing Group gave for this hypothetical. These people all have heard of your service you just began, it was the one thing everyone was waiting for, the next Facebook. Just about two billion now are signing up for this great service, scrambling to get their information into the system, until finally, the accounts start flooding your database. After about 1.3 billion people are populated for eternity inside your database, you notice that the account making start slowing down. You don’t know why, maybe internet in their area is just preventing them from moving on to create their account. But it has been full minutes, approaching an hour, and the accounts are barely trickling in. Those are the people who want to be noticed and have a strong identity. Someone may have taken the name they wanted to use, so they are busy making up another one. Found a good one? Taken. Use one like this with a bunch of unnecessary numbers after it! No one likes to take those random names. And what if this site doesn’t even have recommended names? You just have to guess at what is available, then memorize the random username you have made or use a reference to recreate it. This small problem of non-duplicate names has lost you the business of seven hundred million people. This is why this is such a big problem.

Having a service implies that the service is allowing the user to do anything within the rules of what they have created. If they portray this ideal, but go on to limit the user when it comes to their name on the service, said service is both lying to their users as well as itself as a business. Anyone who can’t get what they want will be upset in any capacity from momentarily annoyed to grudge-holding. As a business, the sole objective is to make a product that the users will enjoy, remember, and come back to for further use, as stated by Joe Putnam (2018). The business is now failing their purpose in this customer’s eyes due to the lack of satisfaction, and will also get a less than stellar review from that same person whose needs have not been met. Of course, if this customer ran into this inconvenience, so did many others. Any time a user decides to try a service, the service runs the risk of that user being unhappy and losing out on their business which Paul Linnell (n.d.) further elaborates on. Ivor Jones (2013) also talks about how satisfied users will only speak to a fourth of the people about the service compared to the unsatisfied population.

At a certain point, the service could stop being practical or useful to the people who have been using it because it has not been changed, a big reason businesses decline, according to Sujan Patel (2017). Once those people leave, the business is now losing some of the revenue and reputation it once had. New users may look into the service the business provides, but when the reviews from previous experiences are more bad than good, the site is slowly declining in popularity, and none of the problems mentioned were fixed, it would be in their best interest to look for something better. This cycle could continue on forever until drastic changes are made to how things are done.

Not only is this a problem for the business as a whole, but the individuals that have had this negative experience are also now troubled as well. Some users signed up after a family friend recommendation, some were the first in their group to do so. Recommending a friend is one of the most useful things to a service, especially when you get perks for doing so, similar to Dropbox, as George Vasiliadis (2018) explains. If a person is recommended to a service, and is denied access, that person will be upset that they are unable to experience what their friend said they would enjoy. Have you ever wanted something, but not had the means to obtain said thing? Short on funds, not eligible to get, or told no by someone with authority over you? That only makes you want it even more. Not only will they yearn for using the service now, but they could be left out of some of the things the other people of the social circle are doing.

This can even create a problem for the friends of a user that is not a part of the service. When a task is needed to be completed, one of the first things that is taken into consideration is the ease of the task. Wouldn’t it be easier to send an email to all department members rather than talking to each individually or in small groups once in person? That is the same idea here, but instead of emailing the whole department, it would be to use the service provided. Sure, you can reach a large portion of the people you need to if there is only one person that is not on the service, but why not only make it one task instead of two? Going out of the way to first use the service, then still have to use other means to contact the person who is not included is a hassle. Personally, the easiest would be to just use the service and be done. To circumvent this whole two part process, moving on to a different service would be better in the long run, but finding that service that provides what you want is more work than it should be.

So far, this has been all about being denied a username. What about the people that do get their chosen name? What if they have grown and no longer like who they are represented as? In the system that was built on the service, the only way to become someone else would be to make another account, which in most cases requires another email address. Although it is suggested by Joelle Alcaidinho (2012) to have four emails, you want the separation of tasks to remain consistent. If you begin to use your spam email to sign up from a service, you might miss important messages in the sea of unwanted notifications. Even if you do have another email you can use, who’s to say you can get a username you want? The main idea here is that you are contracted, in a sense, to the username you claimed. No one else may have it, and you cannot give it up.

The solution to this flaw in the system can be quite tricky. If usernames are required to be unique and have a unique email bound to them, it’s not like someone can just make the same username in a different email or make a new account with the same email. The only way to allow someone to have the same username is to just take that person’s information out of the database, allowing it to be reused elsewhere. But then, the business must go through, change the service to make deleting the account an option, and then update the database to reflect the changes. Once that is done, no one even knows that username is now available. Also, there is no reward for the user. This user was early to the service and was able to snag a username that everyone else wanted. Why should something so sought after go for free?

The solution is a bidding site, similar to eBay in a way, named Sinomim,where usernames are displayed and have bidding wars based on the criteria specified by the user owning the name currently, or can flat out buy the name. Not only does this give a way to release the individual from the contract that is their username, but it also gives them value to do so. As for the bidders/buyers, most of them would be just searching for the right name to use on the service, scrolling through many options that are listed upon the site. There could also be people that already own accounts, but are looking for a better name to use. This site will allow more users to come into the service, give people more reason to stay on the service, due to more connections, and keep the site relevant in their minds. In simple terms, more use of the service, more revenue for the business.

This technology will collect information from the services that have partnered about users, using that information to determine multiple things. Before even using Sinomim, your most recent activity will be monitored. If any user is inactive for a time agreed with between Sinomim and the partnered service, an email will be sent to the user to tell them about Sinomim. If they are interested, they will accept the invitation, making an account and specifying if they want to sell or have it be bid away with a timeframe, with a price for each. From there, other users can see it in the list of names and decided if they want to buy it or bid on it during its timeframe. Once the price is paid, the name will be deleted from Sinomim, the account will be deleted from the partnered service, and be immediately remade automatically with the new email of the buyer and a temporary password. From there, the new user has full access to their purchased account in their email.

To complete this operation, Sinomim will use AngularJS for most of the display options, and will have to be able to support any database that the partnered services may have. The server side will use node to hold the data retrieved in JSON objects. (I don’t know exactly how this will all work, so this is all hypothetical)

PROGRAM & TASKS

Program

Sinomim will be comprised of many different languages, frameworks, and libraries to all form a functional website that will allow users to sell their previous accounts for real money. A system to create an account with Sinomim, and store it into a SQL database, so it can be used to purchase usernames from partnered sites will be made in HTML/Javscript. AngularJS will be used to display and filter through usernames and their information. At first, filler data may have to be used due to the lack of permission to user information, but it will soon be populated after. Sinomim will include an auction system, using the Syncs library, where users can view real-time where the current bid is, and any other addition rules made. Once the auction is over, a form will be built to be sent to a user’s email to prompt them to make the payment.

Tasks

* Set up a database with tables to hold user information and username information.
* Set up an AngularJS project that can communicate with the database.
* Allow user to make an account with Sinomim, adding to the user database.
* Allow user to register an account from a partnered site on Sinomim, using it to sell or bid away, adding to the username database.
* Manipulate data from database to display on webpage in a sensible format.
* Allow user to filter or search through data in any way that applies.
* Allow user to choose an entry, and see more details about said entry.
* Allow user to buy/user to buy on an entry, depending on what it is listed as.
* Shift account from the seller to buyer by deleting the account, remaking it with the same username, the buyer’s email, and a temporary password.
* After a bid ends or purchase is made, the buyer will be prompted to enter in their information for purchase, sending it to the seller’s credit if it is approved. If the buyer cannot pay within three days, the seller will be prompted to redo their sale or auction, or take the second highest bid in case of the auction.
* (Stretch) Users can see a large list of all available names current with any partnered site.
* (Stretch) Users can filter through available names to find anything they would like.

BUDGET

For Sinomim to be built, there is no cost but time associated. The resources that will be needed and used are a Neumont provided SQL server and a space to work.

TASK SCHEDULE

* Task 1 – Think of all the fields needed in the database – 2 hours
* Task 2 – Figure out how to set up a SQL server with Neumont – 3 hours
* Task 3 – Set up SQL server database – 2 hours
* Task 4 – Finalize all features of Sinomim – 3 hours
* Task 5 – Brainstorm all the pages needed for all functions and features of Sinomim – 3 hours
* Task 6 – Design all the pages/modals that will be displayed all throughout the service – 18 hours
* Task 7 – Ask for feedback on each page – 4 hours
* Task 8 – Revise pages based on feedback – 3 hours
* Task 9 – Make the routes and HTML files for all pages needed – 1 hour
* Task 10 – Learn how to connect AngularJS to a database – 2 hours
* Task 11 – Enter in filler data into database – 2 hours
* Task 12 – Connect AngularJS to a database – 2 hours
* Task 13 – Make login and registration, connected to Sinomim accounts database – 5 hours
* Task 14 – Create responsive main screen page – 6 hours
* Task 15 – Create responsive ‘Names for sale’ page – 6 hours
* Task 16 – Create responsive ‘Name detail’ page – 3 hours
* Task 17 – Create responsive ‘Purchase name’ page – 3 hours
* Task 18 – Ask for feedback on each page – 4 hours
* Task 19 – Revise pages based on feedback – 3 hours
* Task 20 – Learn how to use Syncs – 3 hours
* Task 21 – Learn how to Socket.io – 2 hours
* Task 22 – Create responsive ‘Bid for name’ page with socket.io and Syncs – 6 hours
* Task 23 – Create responsive ‘Account’ page – 5 hours
* Task 24 – Make site navigation (links go to correct pages) – 5 hours
* Task 25 – Place user authentication in all the places that need it – 6 hours
* Task 26 – Learn IntroJS – 3 hours
* Task 27 – Use IntroJs to create a tutorial for site – 8 hours
* Task 28 – Learn how to send emails programmatically – 4 hours
* Task 29 – Send an email to a user once their account goes inactive, informing them about the site and it’s functionality – 5 hours
* Task 30 – Send an email to the account owner (will be me since not gathering personal information) that their username has been purchased – 4 hours
* Task 31 – Set up a system to email information and a link to a payment form – 3 hours
* Task 32 – Look up how to create a form that will allow users to pay in some fashion – 4 hours
* Task 33 – Make form for payment of the name that was purchased (any purchase will not actually go through) – 3 hours
* Task 34 – Learn how to use most applicable Steam API here (unless I get access from another company to use their real data) – 10 hours
* Task 35 – Use the Steam API to populate database with relevant data – 8 hours
* Task 36 – Go over site again, making sure everything looks correct with new data – 4 hours
* Task 37 – Do a lot of user testing for full site – 20 hours
* Task 38 – Revise anything that gave the users problems – 9 hours
* Task 39 – Prepare presentation for capstone – 5 hours
* Task 40 – Rehearse presentation – 10 hours

EXPERIENCE

One reason I can complete this idea I have for capstone is that I am experienced with estimating what I can do if I try my hardest, and change scope if needed. This has become apparent to me from my many projects classes at Neumont, especially the one I am doing at the same time of writing this. We are taking a working chess application and separating its logic from its display. When we first began this, we started it from the logic, making it control when the display did things. About a week into the project after very good progress, we were told it was nearly impossible to complete. So to combat this, we switched so events from the display would control what our logic held. This is now taking more time than expected, but by splitting up the tasks and working more outside of class to complete the important things. As of right now, we are looking on track to complete this with a few stretch goals as well.

Another reason this capstone is feasible for me to complete is because I have had this idea for a while, and enjoy it. Let me just put this in proportion for you. I played soccer, tee ball, baseball, and chess for a team, as well as basketball and football with friends and family. I have since quit all of these, with most likely under 50 hours total in all of these beside basketball and chess. I simply quit because I didn’t enjoy them anymore. I also have been playing video games all my life. I have put 428 hours into the Borderlands series (at least on PC, way more on Xbox), 704 hours in Overwatch, 1,205 hours in Rocket League, and even more into just Modern Warfare 2 for Xbox. You can see, anything I do is based on how much I enjoy it.

Knowing this, I can say that I enjoy this idea, considering I have had it for quite a while, as well as have made a few iterations of it since its conception. I will put as many hours as I can spare into making this a great application, and hopefully be picked up by a company for its usefulness.

I will be successful also because of my ability to learn new things that I may need. I have been building my base knowledge, which was originally nothing in terms of coding, up to where it is currently, building projects from nothing in multiple languages. The most prominent example if this is when I had to learn how to use C#, a very new language to me, to build the Game of Life. It used algorithms that I could think of in my head, but declaring classes, binding the data to the view, and creating the views was very hard to think of. I had to reference things I have done before to straighten out the logic, get help to create C# objects correctly, all for hours at a time. I ended up actually doing amazing with my game due to my perseverance, and would be something I say I’m proud of, and would show anyone I have done.

Works Cited

Cherry, K. (n.d.). Conformity: Why Do We Try So Hard to Be Like Other People? Retrieved from <https://www.verywellmind.com/what-is-conformity-2795889>

Email notifications for inactive users. (2017, June 28). Retrieved from

<https://community.atlassian.com/t5/Jira-questions/Email-notifications-for-inactive-users/qaq-p/602864>

(n.d.). Retrieved from <https://www.internetworldstats.com/stats.htm>

Current World Population. (n.d.). Retrieved from <http://www.worldometers.info/world-population/>

What a Unique Selling Proposition Really Means and Why Your Business MUST Have One. (2018, July 06). Retrieved from <https://neilpatel.com/blog/unique-selling-proposition/>

Dropbox grew 3900% with a simple referral program. Here's how! – Inside Viral Loops. (2018, June 26). Retrieved from <https://viral-loops.com/blog/dropbox-grew-3900-simple-referral-program/>

Alcaidinho, J. (2012, August 29). Why You Need at Least 4 Email Addresses. Retrieved from <https://www.apartmenttherapy.com/why-you-need-4-email-addresses-176221>