Needa

Miguel Nunez

The California State University of Monterey Bay

CST 499: Computer Science Capstone

Dr. Brian Robertson

July 7, 2022

Table of Contents

Introduction

Name and Description

Needa is a web application where professionals can post their services. These services can range from a variety of things, for example, carpentry, web development, barbering, etc. The purpose of this is to help create jobs and to make it easier for people to find professionals who offer these services in their area. Needa also strongly encourages entrepreneurship by offering professionals a platform in which to post their services on.

End users

End users will come in two forms: professional seekers and professionals. Professional seekers are those who are in need of services and professionals are those who offer services. Although their purpose for using the application will be different, their experience will be similar. Indeed, both will have the ability to create a profile, search for services, and create live feed posts. The reasoning for this design is that it leaves the door open for professional seekers to both search for services and post their services, if any, and vice versa, professionals can both post their services and search for services from other professionals.

Needa will be made available to the world in the form of advertisements, particularly on social media outlets like Facebook and YouTube. When the application is complete, I will create a demo video and upload it on my YouTube channel. My channel currently has over five thousand subscribers from all over the world, with new members joining daily. Though the number of subscribers is not high, it's enough to get the application out to a fair amount of people. Ultimately, the number of people who decide to create an account and engage with the app will help me gauge the success of it, and that will determine whether it's worth investing more time and money into it.

Feasibility Discussion

Environmental scan and literature review

After conducting a few Google searches, I came across a website named angi.com a.k.a Angie's List. It's similar to Needa in the way that users can find different kinds of services. According to angi.com, “Angi is your home for everything home. A single go-to for home projects big and small, we match you with personalized solutions, highly rated service pros, and fair and transparent pricing for everything from house cleaning to major remodels.(2021) ” One major feature they offer that Needa doesn't, at least not yet, is working professionals that are contracted directly through them. So, for example, if somebody needs a roofer, they provide a quote for the service and they even send a roofing contractor directly to the customer's door.

Project justification

This project will be useful for anybody who is tired of resorting to Google search for a list of professionals in their area. Hopefully, it will also allow them to build friendships with those professionals so they can have, for instance, a “family mechanic” or “family landscaper” who can do the job at a fair price. Perhaps the ones who will benefit the most from Needa will be the professionals themselves. Needa will give them the opportunity to earn side income or even a full time income depending on how much they are willing to work. It will also help those out who are thinking about starting a business but don't have any customers.

Ethical / legal considerations and mitigation

Fake users using the application for something other than what it is intended for is an ethical concern. For instance, somebody can fake being a professional only to meet up with someone to cause them harm. As a developer, there is not much I can do to mitigate this besides warning users about the potential dangers of doing business with complete strangers.

The only legal consideration that I'm concerned about is, for example, if a professional who is not licensed provides a service. Everything is fine until the person receiving the service is unhappy and decides to investigate if the professional is licensed or not. If they are not, they may decide to sue not only the professional but Needa as a company. One way around this, perhaps, is to create a web page strictly for the purpose of letting users know Needa acts strictly as a middle man and not liable for any stolen property, damages, etc.

Long term life and future enhancements

If it so happens that this application is successful, it will require quite a bit of improvements and maintenance. As of right now, I have been doing my best to put it together but I am no expert in putting an application of this magnitude together. This leads me to believe the applications integrity would not hold up once it's used by thousands of users. Even if it does hold up, it will require a major upgrade to the storage plan it's currently under to ensure it can keep up with the information users input, particularly images. At that point, I would need to integrate some kind of subscription service so the users can help with storage and server costs.

Platform

The main languages used for my application are HTML, CSS, and JavaScript. No framework was used, instead I used the JavaScript run time environment of Node.js to run my back-end. To store user data, I used MySQL combined with an Amazon Web Services (AWS) bucket. The AWS bucket was used to store user images as those files are too large to store in a MySQL table. I also used the services of Mailgun for sending emails. To prevent users from submitting too many requests, I used the Google reCAPTCHA V2 API and to show users a map of their location, I used the Google Embed API. The code itself is stored in a GitHub repository, which is linked to Heroku, where it is being hosted. In order to send emails to any email with Mailgun, I had to host my Heroku application with a custom domain, so I linked it to a domain name that I purchased from GoDaddy.com.

Major Functions

There are five main functionalities the application needs in order for it to serve it's purpose. The first is to enable users to register/login. This is essential because without it users simply will not be able to dynamically interact with the website. The second is to allow users to create a profile with information, such as, full name, profile photo, profession, experience, etc. This information is how other users will determine if they want to contact and ultimately hire this particular professional. The third is to allow users to search for users of a particular profession, for instance, a user might search for a plumber or electrician. The fourth is to allow users to friend or befriend other users. This is important because it allows users to keep a contact list of different professionals they may need in the future. The fifth is to allow users to create, read, update, and delete posts on a live feed. This gives the application a fun twist by allowing users to post what they need done and allows professionals to contact them with a price for which they are will to complete the job.

Usability Test Plan

In order to get feedback on my application, I will be conducting a usability test plan on the five main application functionalities. The testers will include my girlfriend, sister, and brother. Before the test is conducted, I will explain the purpose of the application so they can get an idea of what it is, but I won't be giving them instructions on what to do because I want to see the reaction of a “real” user experience as much as I can. Once they test it on their own and I get their genuine feedback, I will ask them to test each of the five main functionalities just to see if my code holds up. Their feedback along with if they are successfully able to complete the five main tasks will help me determine what I need to focus on from now until my presentation on capstone festival week.

Usability Testing/Evaluation

Tester 1

My girlfriend was the first person to test the application. She looked amazed at the website homepage and mentioned that she really liked it. Her main complaint was that each time she searched for a professional she had to input her location as opposed to the website automatically detecting it. She also thinks I could make the lived feed a lot better by allowing users to create a post directly on the live feed page. Currently, I have it setup to where a user creates their post on one page and views the posts on the live feed on another page.

Tester 2

The second person who tested my application was my sister. She already had an idea what the application was about but she had never used it before. Initially, she looked confused as to what she needed to do but then realized she could search for professionals on the home page. She tried searching for a few different professionals but struggled to correctly input her location. The application only accepts locations in the format of “city, state”, notice the space between the comma and state. However, the program kept indicating she was entering the wrong location because she was typing “city,state”, without a space.

Tester 3

The third and final application tester was my brother. He was able to complete all of the tasks. He complained about not being able to see search results for users in his location when he used the zip code location. Before he tested the website, I had created a few fake accounts with a zip code of 93901, which is in the city of Salinas, but when he searched for a professional in the zip code of 93907, which is also in Salinas, no professionals were showing up. He would have needed to search for the exact zip code for them to show up.

Timeline/Budget

My milestones exceeded those in the proposal, however, they weren't necessarily done in the same order. In week 2 of my proposal, I planned to strictly work on the front-end and I did but half way through week 4, I was unhappy with the overall user interface. I ended up re-doing it and that took me about 3-4 days of work. Precious time that I could have used to work on what

I originally planned for that week. Similarly in week 5, I successfully deployed the application as planned but I was unhappy that I was using a domain name of codefoxx.com, one I had previously purchased for a different website. I decided to purchase needa.online in week 6 and it was a headache switching over to it from the previous domain. Once again, that process took precious time that I should have used for working on the milestone for that week. Because I did spend $1 dollar on purchasing the domain, I would say that my project went over the budget as I wasn't planning on spending any money on it.

Final Implementation

When a user visits the website, one of the first things they'll see is a search bar with some text that lets them know they can search for professionals of any trade in their area. Once a user enters the profession of the service they need and click search, they will be redirected to a different page that contains the results of their search. If there are indeed professionals of that trade in the area, the user will be able to click on the “view user” button next to the respective professional and view that individual's profile. A professional profile includes things such as, slogan, skills, showcase images, social media links, etc. Users can also add the professional to their contact list and send them email messages directly from the application; however, a user must have an account and be logged in order to access these features. If they are not logged in, they will see a pop up modal advising them to register/login in order to access them.

That brings me to the register/login capability of the website. As previously stated, users need to login in order to access certain features. The registration page contains the typical fields of first name, last name, email, password, etc. Once a user registers, they'll be sent a verification link to the email they signed up with. They'll need to click the link in order to verify, they will not have access to their account without doing so first. The application also has reset password capability in case users forget their password.

Once a user logs in to their account, they will have access to other parts of the website, such as the live feed. The live feed is a feature where users can create, read, update, and delete posts about whatever they need done. To do this, users will navigate to the live feed page and create a post using the form right under the “live feed” header. This post will be available in the feed of the county that the user selected when they registered, along with the posts from all other users in the county. To change their county, they can go to settings and update their city, state, zip, and county. Consequently, their post will be removed from the old county and appear in the updated county. To update or delete a post itself, users can click the ellipsis icon in the upper right hand corner of their post.

Discussion

There were so many problems that arose while building the application that I don't know where to begin. One of the main problems was how to send information from the front-end to the server and how to receive information from the server on the front end. This was and is a problem because there are a number of different ways to go about it and there doesn't seem to be a clear answer as to when to use one over another. I found myself trying to figure out how to use one method that works for all situations to keep things simple and neat but I simply could not, so I had to resort to using a variety of methods to get it done.

For instance, in order to send multiple images to the server, I can create a form tag with a post method and simply submit the form with a button of type submit. However, when the user uploads new images and refreshes the browser, the server sends back those images so now I had to deal with images coming from the server as well as new images the user is uploading and/or deleting on the front end. The initial solution no longer works and a complicated fetch call is required. The problem with a fetch call is that it doesn't allow the rendering of Node.js routes, instead it sends back JSON data, which has to be processed on the front end. Because it has to be processed in the front end, it didn't allow me to use success/error messages I had set up for when data gets sent from the back-end. As a result, I had to create a second pair of success/error messages that are used to validate data on the front end. This is just one of many cases where I had to figure out how things worked and somehow put it all together the best way I could.

The experience of building this website has been good and bad. It's been good because I proved to myself that I have the ability to build anything I put my mind to. The bad part is that it takes a long time to figure out how to put it all together. One has to fight for every little feature that is added to the project. For example, if you want to allow users to create posts, you have to figure out how to return and display the current users information along with the information of every other user who added a post. But who's post gets to display on the feed? Any users? Only users in the area? What do you do once a post is old? Do you hide it? Do you delete it? How does a user edit a post? How do they delete it? As a developer, one has to have an answer to all of these questions. That forces one to learn every part of the project, that being the front-end, back-end, database, design, and logic. As a developer, if you lack in any of those areas, you simply will not be getting very far.

The future of Needa might be bright or it may not and I am okay with either of those. One very important lesson that I learned about building this application is that you have to have a plan before you start coding a startup application. I built this application without one. With my coding skills, I can figure it out along the way and while that might be true, I found myself doubting the success of the project as I was building it. The reason being that I had no clear plan about the business side of things. Indeed, I was not sure how I was going to make money off of it. If anything, I was going to lose money on the costs of storing data and the server. I should have thought about all of those things before I wrote one line of code.

References

(2021, March 17). Frequently Asked Questions. https://www.angi.com/faq