Senior Project - Group No Name Yet

[00:00:00] **Everyone:** This is the project presentation for the senior project. It ticketing systems incorporated by the group. No name yet. I am the front end developer David Jorgensen. I'm also a front end developer, Steven Lamb. I am the backend developer, Keelan Brening , and I am the other backend developer, Ricardo Escalante.

Ricardo: So the main purpose of the help desk is to allow various users and companies to create tickets whenever they are having any kind of technical issue. Uh, whether it be, you know, simple software issues, like an application, not launching or, uh, hardware issues, like, you know, a monitor, not working or a printer, isn't printing out a form that the employee needs. Um, They can create a ticket for this issue, which will then be assigned to, uh, one of the many technicians available that will go on to help [00:01:00] that person that, uh, made the ticket. So to solve that issue so that they can then continue on with their day, instead of being stuck, trying to fixation themselves, um, this whole process can easily. Increase the productivity of users and the companies by decreasing the downtime of workers, spending their own time, trying to fix the issue at hand.

Stephan: Our team chose this project as it has qualities of scalability and incorporates different technologies. By scalability I refer to the scope of the project. It can start off with limited complexity can, but can be adapted over time to better suit, large scale needs. Um, so essentially the project can grow in terms of its feature set and its um, backend infrastructure to handle large amounts of users. Um, ticketing software seems. Somewhat [00:02:00] simple, but can actually be quite complex as a feature sets grow. So this project presented itself as, uh, adaptable on the loudest to cleanly divide tasks.

Ricardo: So when it comes to the target audience that we are trying to reach with, um, our help desk it's primarily any business or industry that requires or could potentially benefit from a online customer support system ones that might have a lot of technical issues or issues that are very common. Um, as for market research, we found that a research done by Gartner shows that the spending on customer support software had reached over $48 billion in 2018. So it's a very fast and continually growing, uh, industry that can easily be joined with plenty [00:03:00] of, of profit to be made. Um, plus it also makes sense logically to have a digital support platform, especially because of current times with the, uh, COVID pandemic, a lot of people are working from home. Um, so they don't have. I like it, technician that you might typically find in a traditional workspace at their house, so they can use something like our, it ticketing system to be, uh, linked to someone that can help them fix any types of issues that they're having.

Keelan: So for the project, um, our main project plan was to get a website together that would be able to have a secure login for the users and the um, employees as well as for the employees to create tickets, um, for the users and employee side, also add notes to those tickets and for the customers to be able to view the ticket updates as the texts, push notes, [00:04:00] um, so that they know what's going on. Um, as well as to the chat window to talk with the technician live, if they need any other assistance.

David: The project itself was a full year project. So it started in the fall where we did a significant amount of our research into the type of technologies we would be utilizing. And then over the winter break, we got our framework structure built. And finally, into the spring, we developed our structure and the majority of the features to be used and implemented. Um, the project management for this was an agile structure allowing for fast development pace with a low overhead on management managerial requirements.

Ricardo: So when it came to the backend portion of the project, I was responsible for [00:05:00] the base authentication system, so that, uh, users would be able to. Log in and view any tickets that they had created. This was implemented quite easily with the token system that got generated when it really, the user sent their, uh, log info. And then I also handled, um, the, some of the ticket API routes that the friend would use to, uh, get info about. Various tickets that, uh, the currently logged in user was a part of, or that they had created, uh, this included, um, returning. Tickets based on a specific set of rules and filters such as date, um, whether the ticket was open and closed, uh, this was all handled, um, in the backend via, uh, my SQL queries that then returned the data back to the [00:06:00] front end in a JSON formatted object.

Keelan: During the project, I was responsible for setting up the uh, APIs to pull the ticket notes from the server, uh, and sending it off through, back through, uh, JSON format. I was also responsible with, uh, being able to, uh, upload new notes, um, by sending it to the node JS server to push it to our AWS server the store that. I also handled closing of a ticket, uh, which is just a update SQL an update format to SQL and I was also created a option to create new users. Since new users would constantly be joining the it ticketing system that can only be used by the employee side and they can add on new users as they want.

David: Now for the employee side [00:07:00] of the application, the employee will enter into the application through the login page, would they will then input their login information. In this case, their login information is Bob the password. Then enter into the employee dashboard. The employee dashboard holds all of the tickets, which also includes a pagination feature, allowing the user to paginate to go between different pages, which hold the tickets and input their own specific page. Then it also has a filtering system allowing for the different tickets to only be displayed if they happen to have a certain type of priority. In this case, the priority is all or low. Made hired urgent, and then there's status of open closed, or if to include all tickets. They're also sorters allowing for an alphabetical sort based on A to Z or from Z to A for reverse [00:08:00] alphabetical sort. There is also a date sorting based on date, created from youngest date to the oldest date. And then we have the tickets themselves. Where the tickets will then be expanded up in order to, to show the different notes attached to each ticket done by either the technician or the customer. So the customer has the problem characters. I type aren't showing up, let's update the ticket and state that someone is on their way over to that location in order to investigate what's going on. We then update the ticket and it's sent off to the database, the simple refresh it'll reload the [00:09:00] page, and we see that it has been added into the database. Now imagine that the person came back and that the information displayed to them and the issue has been resolved. So we would then put that into the text box. And then again, it is put back into the database. The employee then leaves through the log out functionality and is deposited back on the landing page.

Stephan: So the customer has a dashboard available to them, um, after they log in. Once they do so they're presented with a list of their open tickets by default. Uh, they can also view their close [00:10:00] tickets in another tab. Um, from here, they can also delete, uh, tickets if they so desire by double clicking on the trash icon and the tickets deleted refresh ticket. Um, they can also open a new ticket. From here a model is showed that contains all the necessary information they need to input. Let's say we're having keyboard issues. Um, with ghost presses, something along those lines hit submit the page directs us to the individual ticket page. From here it shows a chat, um, Box that allows us to communicate with the assigned agent from here you know, we can [00:11:00] provide additional information or agent can provide us with updates and these messages are in real-time using WebSockets. So if we wanted to type something. Hit send messages persist in real time. Go back to our ticket list. Um, can see our new issue opened here. Yeah. And that's the entirety of the customer side of the application from here. They can log out.

David: Now we did have a lot of difficulties with this project. Um, each individual person in our group tended to have their own challenges based on their different assignments personally, on the front end side for the employees and the technicians. A lot of my [00:12:00] problems came from CSS and integration. I had a lot of issues, getting the look and the feel of the webpage to responsively react to the users, um, interactions with it, as well as having the compounded problem of integrating into the backend three node based system. Using our, our Axios implementation for HTML calls.

Stephan: I learned a lot about reacting web development. What it really takes to assemble cohesive systems. One of the big challenges for me was determining where a certain feature implementations needed to take place, as well as ensuring those implementations are not vulnerable to attack.

Ricardo: So one of the issues that I had when working on the backend was trying to get certain SQL queries working, [00:13:00] just how I want to returning the right data. Um, sometimes that can be tricky. It might not return the date, the data that you want, or some data will be missing when it should have been there. So I kind of learned how to debug and fix those kinds of issues. And then also the mixing of front-end and back-end sometimes there were inconsistencies with what front-end had already done and what the backend had already done. Didn't max, uh, didn't match. So those kinds of little miscommunications, uh, had to be fixed as well.

Keelan: Some of the challenges that I encountered when doing this project mainly came down to the querying and node js, uh, had a lot of syntax errors when I would test the actual query itself on, at the SQL server and it would work, but in node JS, it would throw an error. So I spent a lot of time trying to get the formatting correct [00:14:00] and being able to send back the correct data that the front end requested as well as making sure that I'm receiving the data that the front end is also sending. Since that was also one of my, my problems that I was encountering. So for the future of this project, we want to continue working on it and improve a lot of the features that we have right now, as well as add new ones. The reasoning for this is that we want to have a full working project with all the bells and whistles that we would be able to show off to our next employer as a project that we have worked on, which we'll be able to showcase what we spent. A whole year working on as well as the ability to show us that we can work with the group. So let's start off with the backend. So for the backend, me and Ricardo, we want to spend some time refactoring the code and restructuring everything. Uh, just because since right [00:15:00] now it's just all on one page. There's a lot we can do with that, which will make it easier to read as well as it easy to add new features since we'll have a better structure for the backend as it comes for the front end, uh, they want to work on implementing a lot more features. Uh, just so it get more, more information for the user, as well as give more information to the employer. Um, one of the big ones we want to do is actually get the project on a live service. Uh, the back end on the nodeJS server on an actual server and we want to get the front end on a working server too, so that anyone can access it at any time.