Full Stack Web Development in Vue.js

By: Adam Hursey, Cole Calamos, Nate Meyer

Introduction

The NKC Family Foundation (NKCFF) requested a web application to be developed for them that would replace their current document management system. The requested application needed to be secure and provide ease of use for all those who are using this application.

Technology Stack

- Vue.js
- Webpack
- Express
- Node.js
- Babel
- Postgres
- AWS
- JWT

The technology stack above was decided based upon the desire for a simple single page application that could be run inexpensively, as well as securely. We felt that using AWS with Vue.js as our main parts of this project would provide functionality at low costs to NKCFF and using JWT for authentication and security would provide an easy and secure means of authentication.

Objectives

- Access certain company information from anywhere
- Store Documents, Contact Information, Organization information, and Grant Information
- Ability to change what forms are displayed on their non-secured page
- Redesign their public non-secure website

Front-End

For the Front-End we used Vue.js. We did a redesign of their original website and an employee portal so that they could access private information. On the non-secured layer of the website, we prioritized design over functionality. This created a design that was very personalized and provided all necessary information. On the secured layer we prioritized functionality over design an were able to pull in components from the external component library Vuetify to help format the information needed on the secure layer.

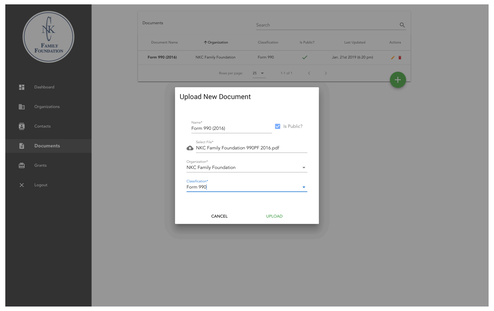

Figure 1: Documents Page

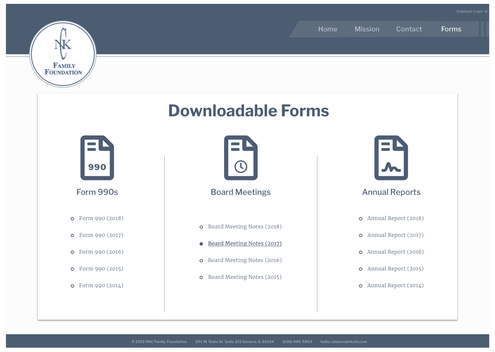

Figure 2: Forms Page

Deployment

For deployment we decided to use Amazon Web Services, due to the large amount of support provided by AWS and since we felt this would be a great introduction to what we will be using in the future. These are the different services we used from AWS: An S3 container to host the static files bundled by web pack for the frontend server; An elastic beanstalk to run the API server on the backend and have it scale up and down based upon the amount of requests incoming; and a RDS server to host our Postgres database.

Back-End

The backend of this project consisted of an entirely different server that would handle as a middleman between the database and the frontend server. The backend was also responsible for authenticating each and every request made to the secured layer of the frontend.

Database

We decided to use PostgreSQL for our database on this project. We had a relational database consisting of 13 tables. The database is only allowed to talk to the API server. It can only receive information from the API server and it can only send information to the API server.

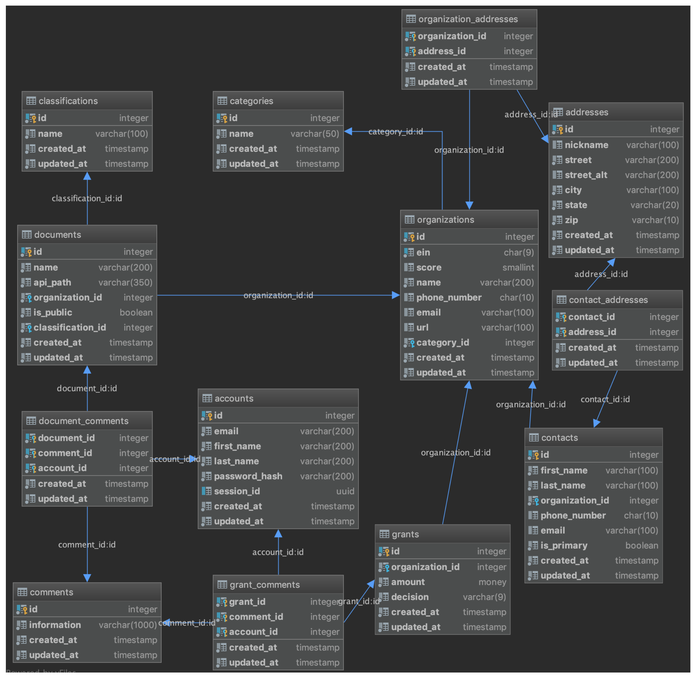

Figure 3: Entity Relationship Diagram

Conclusion

In conclusion we were able to produce a minimum viable product for NKCFF. This application will increase the productivity of members of NKCFF, and it is currently stable enough to be deployed into production. We successfully were able to build this application from scratch. While there is still work to be done we were able to accomplish the objectives that were in the scope of our project timeline.

Future Work

- Accounting Features
- Guidestar integration
- Break components further into abstract functions
- Ability to comment on documents
- Ability to view documents online, before downloading

Acknowledgements

Special thanks to:

- Dr. Jonathan Geisler for proctoring and providing helpful insight into this project.
- Katie Bifoss for being very willing to meet with us and provide answers and clarifications to all and any requirements.
- NKC Family Foundation for providing us with this opportunity to work on a real-world application.