Lowcode: Innovative Web App Development

A Technical Report submitted to the Department of Computer Science

Presented to the Faculty of the School of Engineering and Applied Science

University of Virginia • Charlottesville, Virginia

In Partial Fulfillment of the Requirements for the Degree

Bachelor of Science, School of Engineering

Jaehoon Smith

Spring, 2022

Technical Project Team Members
N/A

On my honor as a University Student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments

Advisor

Rosanne Vrugtman, Department of Computer Science

Lowcode: Innovative Web App Development

CS4991 Capstone Report, 2022

Jaehoon Smith
Computer Science
The University of Virginia
School of Engineering and Applied Science
Charlottesville, Virginia USA
jss3tg@virginia.edu

Abstract

A San Antonio, Texas-based startup that works closely with both commercial and governmental projects required implementation of new web applications and widgets for use in software development of new projects. Mendix, a leading lowcode language, tackles this web application development problem by reducing production time of traditional coding with visual aids to optimize the creation of new websites and widgets. The Mendix language allows users to provide essential websites for prominent companies along with professional skill sets in communication, technical coding, and team building. Combined with React, much of the internship consisted of providing more general developmental tools for the company while also producing customized websites for potential clients. Further development of more widgets and websites are always possible in order to continue production of more client-based projects. With lowcode becoming a rising software development avenue, the next look may be to introduce programs like Mendix as a new leading tool to the coding community.

1. Introduction

When I first heard about Mendix and its emphasis on lowcode, I thought, "How hard could it be?". At first glance, the Mendix Integrated Development Environment (IDE) displays bright and colorful diagrams that heavily focused on the concept of Object Oriented Programming (OOP). However, soon after getting through the basics, I delved into the world of Javascript based custom widgets in Mendix. The task was to build customizable charts based on the Plotly.js library with integration into the Mendix IDE. Consistently updated data called for visualizations that vastly improved the streamlining of client information. As the internship progressed, my next project was to team with an intern group to develop a Proof of Concept (PoC) and a Minimum Viable Product (MVP) website for a potential client. The goal was to showcase a functional application that could onboard a new project for the company.

2. Related Works

When applying the Mendix library to web applications, the documentation is broad yet thorough. Both new and old widgets or import packages are provided with guides to utilize the code and designed to integrate with ease. Plotly (2019), a Javascript library extension, created a funnel chart that allowed integration into Mendix. Implementing this funnel chart widget and the respective blogs/documentation yielded concrete results in processing data, creating new features and optimizing the efficiency of the application as a whole. Furthermore, the integration of Javascript allowed the use

of its libraries which indicated even greater potential for development. Mendix Documentation (2022) also introduces API calls that allow for varying applications to be integrated into a developing web application. As I undertook a client project, I was able to create Jira API calls that served as a feedback system for the client.

3. Project Design

In order to develop the final internship project using Mendix and Javascript, I needed to learn the structure and application of each language. As a sample project guide, I built an airline booking application that utilized the use of data models, associations, and actionable microflows. As the application was finished, I began to research other useful techniques such as widget design and import tools that were used for the final project.

3.1 Widget Development

Working alongside senior business engineers, I delved into the concept of Javascript based custom widgets. The process of creating widgets called for utilizing an imported Mendix library, creating the HTML design of the widget, and implementing the functionality through Javascript. The particular widget that I was interested in making was a funnel chart through the Plotly.js library. The code to develop the widget functionality is shown below in Figure 1.

```
import { createElement } from "react";
import { ListValue, ListAttributeValue } from "mendix";
import Plot from 'react-plotly.js';
import Big from "big.js";

export interface FunnelAreaContainerProps {
    funnelObjects: ListValue;
    textAttr: ListAttributeValue<string>;
    valuesAttr: ListAttributeValue<Big>;
}

const FunnelAreaContainer = (props: FunnelAreaContainerProps) => {
    var funnel = props.funnelObjects.items;
    var arrLabels = new Array;
    var arrValues = new Array;
    if(funnel){
        funnel.map(object =>{
                const textAttrValue = props.textAttr.get(object).value;
                arrLabels.push(textAttrValue);
                const valuesAttrValue = Number(props.valuesAttr.get(object).value);
                arrValues.push(valuesAttrValue);
        });
    }
```

Figure 1: Javascript Widget Code

This code initializes the values taken from the Mendix IDE and iterates through each item to make the funnel chart. The output of the code as shown in Mendix is depicted below in Figure 2.

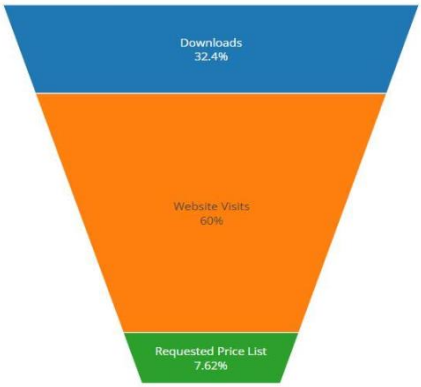

Figure 2: Funnel Chart

The end product displays data inputted from Mendix and is able to visualize data through a cascading funnel with each entry sized to its respective percentage amount. The chart also consists of a legend that highlights individual sections as well as options to edit, delete, or add information.

3.2 Client Project

Now that I had experience in producing Mendix web applications and its extensions, I teamed with a fellow intern to develop a Minimum Viable Product (MVP) for a potential client. The project introduced new challenges as I learned to develop the application under pressures of deadlines and consistent communication with both the team and the client. The application consisted of all of our learned techniques while also required implementation of new features.

To begin the project, Mendix requires a strong data structure that builds upon the backend data management. This management is done through a domain model that consists of entities and associations. Creating an entity is similar to

initializing an object in traditional code. The entity serves as a general piece of data that can be created and altered such as a document or a person's information file. In order to make entities function throughout the project, associations connect entities to display a hierarchical breakdown between entities. For example, if there is a teacher entity, there may be multiple student entities that may be correlated with a teacher. This process was used to initialize the key entities of the project. An example of a typical domain model is depicted below in Figure 3.

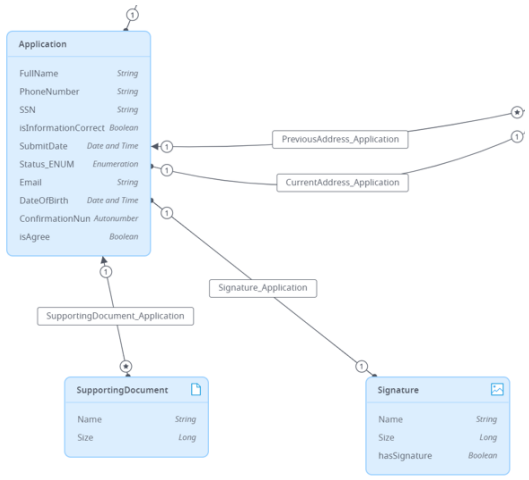

Figure 3: Domain Model

3.3 Challenges

At first, my time consisted of head scratching and scouring the Mendix documentation for insights. Some moments felt impossible as I tried making sense of the entire development framework I had been exposed to while planning each little step. As project scales got bigger, the business aspect of talking to clients, demoing every new feature, and adhering to deadlines became arduous yet influential for professional growth. However, much like learning any new skill, practice and pressure slowly improved my work ethic and abilities to be a prominent developer. Given the mentors and peers, I also had a support system to call for help on any new assignments that I had trouble with. Overcoming these vital challenges was an essential skill for a new developer like myself.

4. Results

The new widgets that I had created allowed for much faster implementation. For the funnel chart, rather than creating a new graph, filling individual fields, and optimizing size/colors, the widget was a simple click-and-drag object that automatically filled with desired entities. The web application project was an immense speed up as work that had been previously done by hand was now completely automated to store and process information.

5. Conclusion

In conclusion, the internship consisted of meaningful and tangible projects that contributed to the improvement of company growth. The completion of widgets provided for faster intake and display of data. With the introduction of a potential client, I was also able to develop a web application that aided in onboarding a new project for the company. Through these projects, I improved my own personal skills in programs such as Mendix and Javascript.

6. Future Work

My experience gained at this company can be extended as I had also worked on other widgets that had not been finalized. If the client has decided to onboard, I could provide further implementation of more features to the application that could eventually be pushed for a final product. The continuation of these works would show the varying use cases of Mendix and make the world of lowcode languages widely known as an alternative method to traditional coding.

References

[1] *Plotly (2019) Funnel and Funnelarea Charts in JavaScript.* Plotly. Available at: https://plotly.com/javascript/funnel-charts/ (Accessed: December 2, 2022).

[2] *Mendix Documentation* (2021) *Service Exposure*. Mendix. Available at: https://www.mendix.com/evaluation-guide/app-capabilities/service-exposure/ (Accessed: December 2, 2022).