CS-499-T4251 Computer Science Capstone

Professor Goggin

Carlos Aguirre

Southern New Hampshire University

March 19, 2023

- **Software Design and engineering**

For this first category as part of my final project, I decided to work with an artifact that was completed during **CS-360 Mobile Architect & Programming.**

Artifact: Event tracking App – Built in Java and Android Studio**.**

The artifact, which was an Android Studio and Java-based event tracking app, was developed as part of the CS-360 Mobile Architect & Programming course. The app offers CRUD actions for adding, removing, updating, and deleting Events. To facilitate user account creation and data management, the app was additionally linked to a database.

I chose to include this in my ePortfolio because I wanted to entirely rebuild an application using Javascript, another computer language. In addition, I can demonstrate my abilities as a full-stack engineer, a skilled software engineer that develops, tests, and implements a wide range of software applications. The artifact was enhanced because it is now functional and I demonstrated additional abilities in front-end languages and frameworks (HTML, CSS, JavaScript), back-end technologies and frameworks (NodeJS, ExpressJS, Django, Flask), database management systems (MySQL, SQL SERVER and PostgreSQL and MongoDB), version control, and web hosting platforms.

Enhancements:

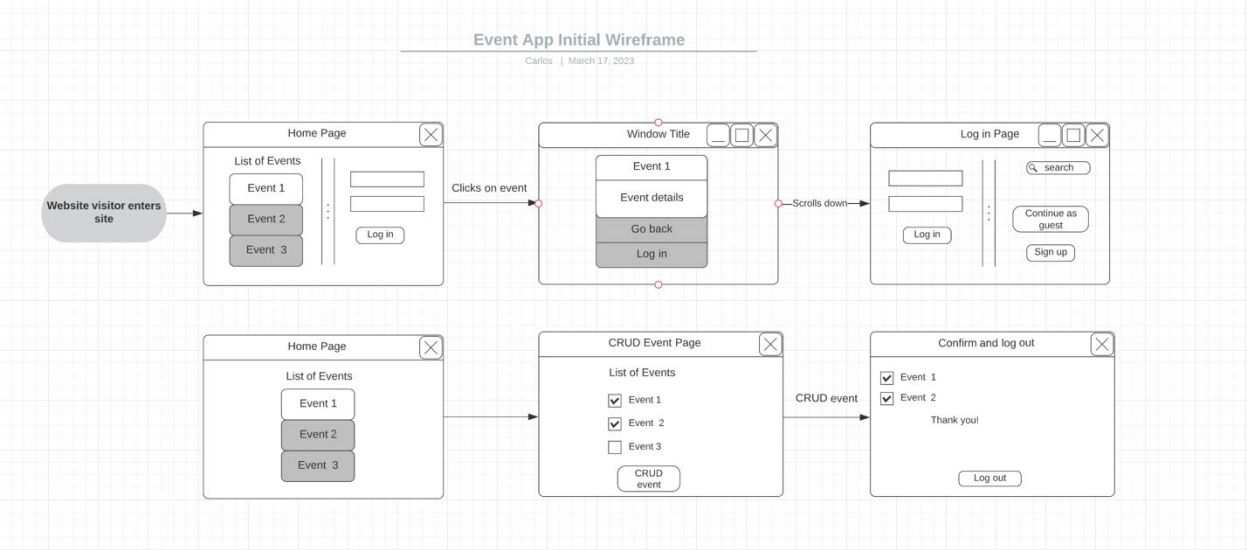
Software design is a process that converts user requirements into a useful form, assisting the programmer in the creation and implementation of software. It has to do with converting the client's need from the SRS (Software Requirement Specification) paper into a format that can be quickly implemented using a programming language. (“Software License Types Explained: What You Need to Know”)

The first step in the SDLC (Software Design Life Cycle) is the software design phase, which shifts the focus from the problem domain to the solution domain. In software design, the system

is viewed as a collection of parts or modules with distinct behaviors. (“Software License Types Explained: What You Need to Know”)

As stated in my enhancement plan, the objective is to construct a web application, where I have been employing Javascript, HTML, and CSS. With the help of the open-source, cross-platform runtime environment Node.js, programmers may build a wide range of JavaScript-based server-side tools and apps.

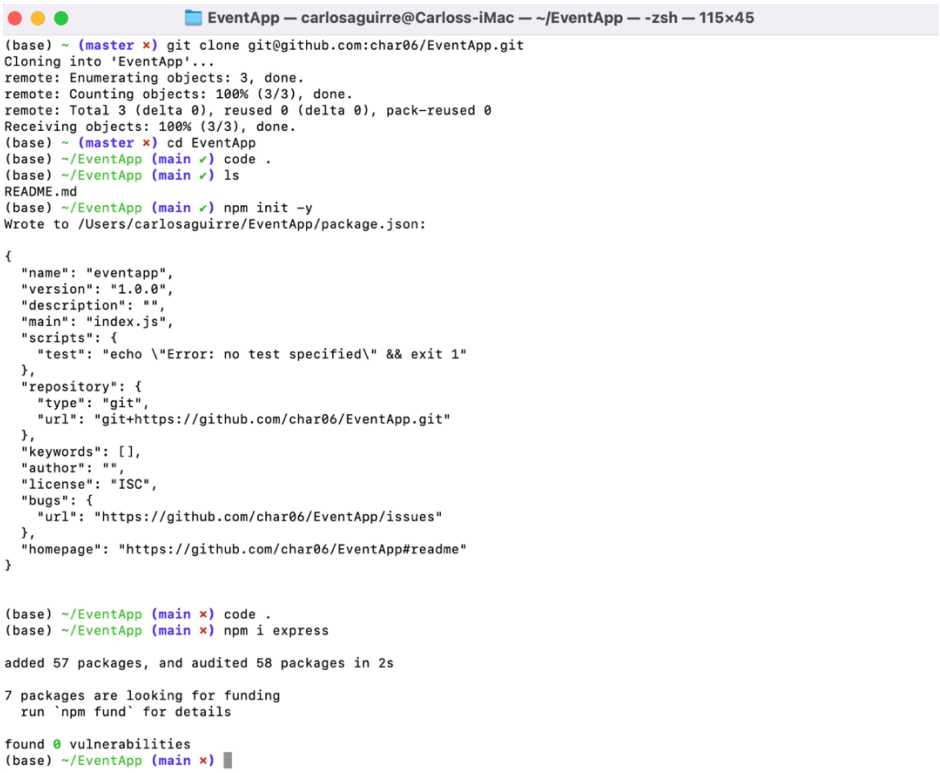
Here is the web application's first wireframe that I made. Wireframes are widely used to organize content and functionality on a website while considering user wants and user journeys. Wireframes are used early in the development process to define the basic structure of a page before adding visual design and content to it.



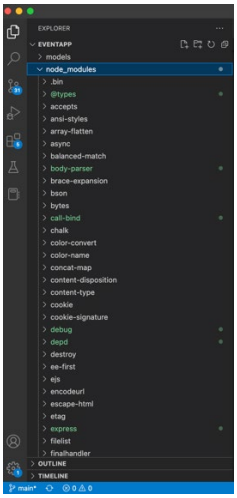
Additionally, there are some typical web development jobs that Node itself does not natively handle. Node won't be very useful on its own if you wish to add specific handling for various HTTP actions (such as GET, POST, DELETE, etc.), handle requests at various URL paths individually (referred to as "routes"), serve static files, or use templates to dynamically construct

the answer. Either you'll have to create the code yourself or a web framework will save you from having to do it from zero.

I started building the application by creating an express app, as you might know node.js is JavaScript runtime environment that is fast, but it does not support request handling, HTTP methods or serving files. So that's why I created the express app first. The reason why I decided to use the express framework was to speed up the development process.



After express has been created, all its dependencies are visible in Visual Studio.



With the assistance of EJS (Embedded JavaScript), a template engine, I used templating. It uses JavaScript syntax; therefore, I choose to use it. It allows us to insert JavaScript code, as the name implies, into a template language that produces HTML.

Let's not forget that software design relates to the modules and components of the software, whereas software architecture refers to the overall structure into which these modules and components fit. Therefore, I made the decision to begin these improvements by building the program's essential features. This involved setting up the server, connecting to the database—in this case, mongo DB—and beginning to construct the user interface's logic while interacting with the server and the database using EJS, JavaScript, and HTML.

Following picture shows the successful connection with the server:

In addition to using Mongo db to build the database, I also used Mongoose, a Node.js-based object data modeling (ODM) module for MongoDB. Like an Object Relational Mapper (ORM), such as SQLAlchemy for conventional SQL databases, it is used. Having the ability to impose a certain schema at the application layer is the issue that Mongoose seeks to address. Mongoose supports a range of hooks, model validation, and other capabilities in addition to enforcing a schema that are intended to make working with MongoDB easier.

Event-App is the name of the database that was built, and it was verified using a connected terminal. The database was then seeded after that. This is done in order to begin constructing and testing the CRUD activities within the web application and to ensure that they are functioning properly.

```
Node.js v19.4.0
[nodemon] app crashed - waiting for file changes before starting...
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
Serving on port 3000
Database connected
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
Serving on port 3000
Database connected
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
Serving on port 3000
Database connected
^C%
(base) ~/EventApp (main ×) git status
```

Database seeding refers to the process of initial data seeding a database. During installation, a database is "seeded" by providing it with a starting set of data. With the next update, I will be able to provide more information about the algorithms and data structures that have already been built to complete what I have so far. I am still lacking styling, user authentication, and maps, which are part of the enhancements. A little video can be viewed at the following link to get a better idea of what has been accomplished thus far.

Quick Video with updates:

https://www.dropbox.com/s/1kyy3g8wx1oxfwp/Fri%20Mar%2017%202023%205_44_21%20PM_default_e5390461.mp4?dl=0

Improving the artifact has been enjoyable, and even if the application is not yet finished, I enjoyed examining the software requirements and planning how to complete what I had built so far while considering modern software engineering approaches. Working closely with the technology's online documentation was something I quickly picked up. To speed up the development of the application's front end, for instance, I chose to use EJS. But software professionals are not always able to recall every little nuance of how frameworks or programming languages operate.

What we can do is to use the resources that we have available for example the documentation online for different technologies. I did use Mongo DB documentation as well as EJS and mongoose.

The only difficulty I've had is time; working a full-time job sometimes prevents me from making things even better, but I believe that time management is crucial, so I've been trying to turn in my assignments early to receive feedback and, if required, make improvements.

References

"Software License Types Explained: What You Need to Know." *Www.ivanti.com*, www.ivanti.com/blog/software-license-types.