# CS-465-Fullstack-Development-I

CS-465 Full Stack Development I with MEAN

# Architecture

- Compare and contrast the types of frontend development you used in your full stack project, including Express HTML, JavaScript, and the single-page application (SPA).

For the front-end development of our full-stack application, we used Express HTML to build a customer-facing page. The customer-facing page uses a Model View Controller (MVC) design pattern to route user requests, retrieve the view for the specified route, and render data from the server database for each specific view. When using the Express application each user action triggers a full page reload. For the client-side front-end development, Angular was used to build a single-page application (SPA), which uses components, services, and routes to load a single HTML page on the client side, and it dynamically updates the content of the page as the user interacts with the application. This initial load only happens once, and the page simply updates as changes are made to any of the data. One key difference is that the Express application may contain several pages, whereas the SPA is what its name says, a single page. In an SPA, components are used to render various parts of the UI and help keep the application’s structure more organized. JavaScript is used throughout the front-end and back-end, enabling developers to code the entire application using the same language. This allows for some of the application logic to be moved from the back-end to the front-end.

- Why did the backend use a NoSQL MongoDB database?

MongoDB stores data (documents) in JSON format, which is essentially a JavaScript way of holding data. This fits well into our JavaScript-centric MEAN stack application since both the front-end and back-end are coded using JavaScript. MongoDB provides a balance between the speed of pure document databases and relational databases, and the data is stored in JSON format making it an excellent method for storing data for the MEAN stack.

## Functionality

- How is JSON different from JavaScript and how does JSON tie together the front-end and back-end development pieces?

JSON, short for JavaScript Serialized Object Notation, is a format for storing data that is lightweight, whereas JavaScript is a programming language with a much wider range of capabilities. JSON ties together the front-end and back-end development pieces because data can be passed back and forth from the front-end to the back-end with JSON objects. Both the front-end and back-end know how to handle these data objects, which helps to simplify the interaction between the two.

- Provide instances in the full stack process when you refactored code to improve functionality and efficiencies and name the benefits that come from reusable user interface (UI) components.

One instance where code was refactored was using the handlebars templating engine for the /travel route on the customer-facing travel page. Instead of having to write the HTML for each individual trip, we can use a handlebars helper (#each) that will iterate over all the trips in the trips database and render each trip’s data. This eliminates the need for redundant coding and simplifies the amount of code needed to render each trip in the browser.

Also, on the client-facing SPA, we are able to use a single component, in this case, the trip-listing component, that displays an add-trip component, as well as all of the trips that are in the database. Again, this iterates over all trips that are returned from a call to the trips database and displays each trip to the trip-listing component. To keep the data more organized, another component is used called trip-card, which essentially tells how all of the data for each trip should be displayed, and if each trip-card has other components, such as an edit or delete button, they will be displayed along with each trip’s data.

This is beneficial because it eliminates the need to code for every single trip in the database. We can use reusable interface components, such as the trip-listing and trip-card components, which are coded one time, and can be used as many times over again as needed. A key benefit of this is it saves time and resources.

## Testing

- Methods for request and retrieval necessitate various types of API testing of endpoints, in addition to the differences of testing with added layers of security. Explain your understanding of methods, endpoints, and security in a full-stack application.

In full-stack applications, methods typically refer to HTTP methods, such as GET, POST, PUT, and DELETE, which are used in RESTful APIs. Endpoints are the specific paths in the API that correspond with various functions, and each endpoint is used with a specific HTTP method. For example, in our full-stack application, we have the endpoint “/api/trips” that uses the GET method, and when accessed, it returns a list of all trips in the trips database. We can also add a tripCode string to the end of this to use another GET method to return a single trip, such as “/api/trips/MEGR220119”. We also have additional endpoints that are used for logging into the SPA and registering a new user. The login endpoint is essential for the client-facing application, as it allows a means for the client to log into the SPA where they are able to modify data that is stored in the database. Security is essential in the application because we do not want just anybody to have access to the SPA and be able to make changes to the database or potentially gain access to sensitive information. In the case of this application, if a user successfully authenticated through the login API, then they gain access to data in the database.

## Reflection

- How has this course helped you in reaching your professional goals? What skills have you learned, developed, or mastered in this course to help you become a more marketable candidate in your career field?

For me, a major stepping-stone in my degree program has been reaching the full-stack classes where I knew I would be learning how to build a full-stack application. I am currently unsure exactly what field of Computer Science I want to land in for my career, however, full-stack development is an area of interest. In this class, to help me become a more marketable candidate in a career field, I have learned how to build a full-stack application with both a customer-facing site using Express and Node.js, and a client-facing application using Angular.js, all of which incorporate the use of a NoSQL database using MongoDB. I learned how to make use of endpoints that can be tested using HTTP request methods, how to incorporate security through the use of authentication and Java Web Tokens (JWTs), and how to render data both on the customer and client-facing application.