**Project proposal: Product Datasheet Search Website**

**Stack Focus**

* Frontend (Web): Route handling through the server-side logic, html and stylesheets loaded from the backend
* Backend: Node.js with Express: For handling API requests, server-side logic, and integration with the database. The project will be mainly focused on the backend of the application. However, there will be a front-end UI that is easy to understand and use.
* Database: MongoDB Atlas and Compass: Not only will I use this to store and manage the data, but I may also use it to create my own Rest API

**Project Type**

* Website: A web application accessible through modern browsers.

**Project Goal**

* The goal of the project is to make a user-friendly UI for our customers to get technical data and download pdf datasheets.

**Potential Users**

* All existing customers will benefit from having access to the technical data of our product. When using our product, it is sometimes necessary to verify dimensions or run parameters.
* New customers will be able to access datasheets in order to check if our product will support their needs within their machine tool environment.

**Data and API**

* Data: The data is comprised of technical metrics in the form of numbers and strings. There are also supporting images that will pair with the technical data.
* Data Collection: The data is already available through the manufacturer’s website. I will be pulling a sample of data to create an API.
* API: This will be created by me, using the manufacturer’s data.

**Project approach**

* Determining the database schema – NoSQL collections. Each part number will have a set of metrics and an image as endpoints that will be linked to the html document. Schema example is below.
* Sourcing your data – I will create my own by using the publicly available documents
* Determining user flow(s) – There will be a form field for customers to input a part number, or they can choose from a drop-down menu in order to get a list of results displayed. From the list of results, the customer should be able to select an individual record to display the technical data of that record.
* Setting up the backend and database- Using methods available on the schema, I should be able to access individual files or a collection of files based on the user input on the frontend.
* Setting up the frontend – The front end will have a simplistic look to it by design. Therefore, I don’t anticipate needing to use a framework like React. I also think most of the data about routes will be rendered on the server-side.
* What functionality will your app include?
  + User login and sign up
  + Users should be able to save datasheets by clicking a button and having their collection stored in their user profile.
* Stretch goals
  + Connect each data sheet to a machine that it is compatible with
  + Creating a separate collection of machines to link the datasheet in a query

**Data Schema:**

datasheetSchema = new Schema({

    partNumber: {

        type: String,

        required: true

    },

    description: {

        type: String,

        required: true

    },

    rpmOutput: {

        type: String,

        required: true

    },

    rpmInput: {

        type: String,

        required: true

    },

    torqueOutput: {

        type: String,

        required: true

    },

    img: {

        type: String,

        required: true

    }

})

userSchema = new Schema({

    email: {

        type: String,

        required: true,

        unique: true,

        lowercase: true

    },

    password: {

        type: String,

        required: true,

        minLength: 8

    }

})