DATABASE DESIGN

Carina Sylvester
SWDV-691

I am using **MongoDB** for my database.
Since MongoDB is a schema-less NoSQL database, it allows me to insert data without any predefined schema and gives me the ability to make changes to the format or data model at any time without application disruption. NoSQL databases don't require nearly as much preparation as relational databases, and they are great for horizontal scaling. Scalability is important to me because I want to continuously implement more advanced features.

In addition, I am going to use the ODM library **Mongoose** that manages relationships between data, provides schema validation, and is used to translate between objects in code and the representation of those objects in MongoDB.

Below are Mongoose schemata I will be implementing:

```
const userSchema = mongoose.Schema({
  id: {
    type: Number,
  },
  name: {
    type: String,
    required: true,
    match: [/^[a-zA-Z\s]*$/]
  },
  email: {
    type: String,
    required: true,
    match: [/^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$/]
  },
  password: {
    type: String,
    required: true
  },
  assets: {
    type: Number,
    required: true,
    min: 0
  }
});
```

The **User** schema includes a user ID; the user's name -which can only include upper & lower-case letters (match); the user's email address and chosen password; and the amount of assets (money) they hold.

This data will be entered into the database during the registration process and is used for verification purposes during the login process. Email address and password can later be changed by the user, and the number of assets (money $) will change based on transactions and

The **Stock** schema describes the stocks on the stock market available for purchase. The ticker is the unique series of letters assigned to a stock for trading purposes (e.g. Apple = AAPL, Tesla = TSLA). Exchange is the exchange the stock is trading on; initial price is the price that a stock in the user's portfolio was bought at; current price is the current price of the stock; description includes background information on the stock/company; and favorited shows whether this particular stock was favorited by the user or not.

```
const stockSchema = mongoose.Schema({
  id: Number,
  ticker: String,
  exchange: String,
  name: String,
  initialPrice: Number,
  currentPrice: Number,
  description: String,
  icon: String,
  favorited: Boolean
});
```

```
const purchasedStockSchema = mongoose.Schema({
  userId: {
    type: Number,
    required: true,
  },
  stock: {
    type: mongoose.Schema.Types.ObjectId,
    ref: Stock,
    required: true,
  },
  ticker: {
    type: String,
    required: true,
  },
  shares: {
    type: Number,
    required: true,
    min: 0
  },
  amountInvested: {
    type: Number,
    required: true
  }
});
```

The **Purchased Stock** schema incorporates the Stock schema, as well as the number of shares purchased by the user (shares) and the amount of money used in the purchase (amountInvested).

The **Transaction** schema includes the transaction type (buy/sell); the number of shares bought or sold; the amount of money added or subtracted to the user's assets (investment); and the date of the transaction.

```
const transactionSchema = mongoose.Schema({
  userId: {
    type: Number,
    required: true,
  },
  transactionType: {
    type: String,
    required: true,
  },
  tickerBought: {
    type: String,
    required: true,
  },
  shares: {
    type: Number,
    required: true,
  },
  investment: {
    type: Number,
    required: true
  },
  transactionDate: {
    type: Date,
    default: Date.now
  }
});
```