**Wet Bat Travel Agency – Quoting Feature: Documentation**

1. **Architectural/design decisions:**
   1. Multi-page application:
      * Easily scalable.
      * Separation of concerns.
   2. ElephantSQL
      * Cloud hosting for security.
      * Easily scalable.
      * Simple to deploy.
   3. Node/Express middleware
      * Easily compatible with React FE (written in JS).
      * API for routing is modular.
      * Simple db setup/connection using node-pg.
      * DB reset setup for development – simple creating/deleting
      * tables and seeding/reseeding data.
2. **Database design**

Please see ERD in root directory. The logic is as follows.

* Many customers who need quotes, therefore a customers table is created.
* There are many quotes, therefore a quotes table is created.
* One customer can have many quotes, and one quote can have many customers. Therefore, the relationship between customers and quotes is many-to-many.
* A join table must be created connecting customers and quotes because of the many-to-many relationship. The join table must contain a customer\_id and a quote\_id for each entry.
* The join table must also contain another customer\_id (called booker\_id) to identify the person who booked the group of people.
  + - * Example: the mother of a family books her, her husband and 3 children. Each family member is assigned a unique customer\_id, but all the same quote\_id. All of the entries in this join table will have the same booker\_id (for this trip), which will be the customer\_id of the mother.
* There can be many kinds of transportation methods for each quote. Therefore, the relationship is one (quote) to many (transportation), which means a transportation table must be created.
* The count of people per quote can be calculated using an aggregrate SQL function and therefore does not need to be a column in any table.

1. **React components**

**Navigation**

* 1. Nav.jsx
     + Top nav.
     + Contains link to home page (main logo).
  2. SideNav.jsx
     + Side nav.
     + Contains link to home and quotes page.
     + Receives props:
       1. selected: determines which nav item gets set to selected CSS.
       2. setSelected: onClick method that changes selected value to clicked nav menu item.

**Quotes Page**

* 1. Quotes.jsx
     + Page component; navigate here from clicking on SideNav link.
     + Parent component to QuotesList.jsx and QuotesView.jsx
     + Two axios calls in useEffect:
       1. GET request for all quotes.
       2. GET request for all transportation.
     + State:
       1. quotesData: all quotes data from database.
       2. transportationData: all transporation data from database after cleaned from transportationHelper function.
       3. currentQuoteID: set when user clicks on quote in QuotesListItem.jsx.
       4. deleting: sets deleting loader when user deletes quote in QuotesView.jsx.
       5. triggerCount: triggers useEffect to make axios calls to query updated data.
     + sortByTime and sortByPrice: functions used to sort quotes in QuotesList.jsx.
     + confirmDeleteQuote: function used to delete quote once user selects delete button in QuotesView.jsx.
  2. QuotesList.jsx
     + Child component of Quotes.jsx; parent component to QuotesListItem.jsx.
     + Props:
       1. quotesData: as described in Quotes.jsx.
       2. sortByPrice and sortByTime are functions as described in Quotes.jsx.
       3. setSelectedQuote: function used to set CSS for selected quote in list.
       4. currentQuoteID: as describeed in Quotes.jsx.
       5. deleting: as described in Quotes.jsx.
     + quotesListItems: returns a QuotesListItem component for every quote in quotesData.
     + Price and time buttons allow user to sort by price and time of quote.
     + Deleting boolean triggers the loader after quote is deleted from QuotesView.jsx.
  3. QuotesListItem.jsx
     + Child component to QuotesList.jsx.
     + Displays each quote’s name, departure date, destination and price for a quick view.
     + Quotes are clickable and more details will appear in QuotesView.jsx component when clicked: id is passed down as a prop to pass into the selected quote function.
  4. QuotesView.jsx
     + Child component to Quotes.jsx.
     + Displays the details of the quote that was clicked from QuotesListItem.jsx
     + Delete function allows users to delete quote from DB – first confirm delete by transitioning to confirmDelete state.
     + Error handling if quote is not clicked in QuotesListItem.jsx and delete button is clicked.

**Home**

1. Home.jsx

* Page component; navigate here from clicking on SideNav link.
* Hardcoded main card that shows welcome banner.
* Parent to HomeQuickQuote.jsx.

1. HomeQuickQuote.jsx

* Child component to Home.jsx.
* In development, but will allow user to create new quote.

1. **Helper functions**
   1. dollarsConverter:
      * Converts price\_cents to human readable dollars for display purposes on front end.
   2. humanReadableDate:
      * Converts SQL datetime to human readable date for display purposes.
   3. transportationHelper:
      * Allocates all transportation data into an object, where the keys are the quote\_ids they belong to and the values are arrays of transporation modes.