

WET BAT Travel Agency : Documentation

Architecture/stack of technologies:

FRONT END (React-app)

1. Libraries

- [React](#) for Front End
- [Material UI](#) for some components design. Icons
- [Formik](#) – for Form submission. Formik allows to handle most of state management of any form. Especially useful when application has many submission forms.
- [Yup](#) – for Form data validation, verifies type of data before passing to BE
- [React router](#) – routing to component/page
- [Node-sass](#) as dev dependency. Variables used for CSS styling (single source of truth). For example, if designer decides to change particular color, it can be changed only from one place.

2. File structure

- Files divided in 3 major sections, 1) Navbar, 2) SideMenu, 3) Main.
- Main folder has subfolders organized according to Components rendered on each page. 1) HomePage, 2) Quotes
- Quote components consist of 3 main subcomponents 1) CreateQuote, 2) QuoteList, 3) QuoteDetails
- Each quote Subcomponent consist of simple design components/Atoms/Molecules

The screenshot shows a 'Quick Quote' form with a light blue header and a white body. The form is organized into four columns. The first column contains three dropdown menus for 'FROM', 'DESTINATION', and 'TRANSPORT', each with a 'Please select' placeholder. The second column contains two date pickers for 'DEPART DATE' (05/11/2020) and 'RETURN DATE' (dd/mm/yyyy). The third column contains two text input fields for 'FIRST NAME' and 'LAST NAME', both with red error messages 'This is a required field'. The fourth column contains two text input fields for 'PHONE' and 'EMAIL', both with red error messages 'This is a required field'. At the bottom left, there is a green 'CREATE' button with a right arrow. At the bottom right, there is a 'CURRENCY' dropdown menu set to '(CAD)' with a 'Please select' placeholder. A small icon of a person is in the top right corner.



FROM	DEPART DATE	FIRST NAME	PHONE
Please select	05/11/2020	This is a required field	This is a required field
DESTINATION	RETURN DATE	LAST NAME	EMAIL
Please select	dd/mm/yyyy	This is a required field	This is a required field
TRANSPORT	PRICE		CURRENCY
Please select			(CAD)







CREATE >


Please select

3. Functionality

- **SideMenuBar** renders components based on selected Menu Item
- **UseEffect** uses **async/await** model with axios. Additionally, **try/catch** for handling errors.
- **QuotePage** component makes axios calls to fetch data of all quotes and all options used during quote creation, such as list of available transports, list of cities, etc
- **QuotePage** component endpoints:
 - i. "GET /api/quotes"
 - ii. "GET/api/quotesdata"
 - iii. "POST /api/quote"
 - iv. "PUT /api/quote/id"
- **CreateQuote** controls that required field must be provided, otherwise it will display warning (implemented with Yup). Created quote immediately appears in the list of quotes
- **QuoteList** is clickable, consisting of **QuoteListItems**. If any quote is clicked additional form will be rendered, to allow EDIT.
- The selected quote passed to QuoteDetail component filtered from the list of quotes, based on selected quoteId
- **QuoteDetails** render form containing all information and values are set as initial values, allowing to change them if needed.
- Upon submission the QuoteDetail component will disappear if BE confirms success
- Material UI for some components design. Icons
- Formik – for Form submission. Formik allows to handle most of state management of any form. Especially useful when application has many submission forms.
- Yup – for Form data validation, verifies type of data before passing to BE
- React router – routing to component/page
- Node-sass as dev dependency. Variables used for CSS styling (single source of truth). For example, if designer decides to change particular color, it can be changed only from one place.

 **Quote Details** 

FROM (SVO) Moscow(SVO)  <small>Please select</small>	DEPART DATE 30/11/2020 	FIRST NAME Alice	PHONE 555-1111
DESTINATION (SVO) Moscow(SVO)  <small>Please select</small>	RETURN DATE 30/11/2020 	LAST NAME Alison	EMAIL a@gmail.com
TRANSPORT Plane  <small>Please select</small>	PRICE 1200	CURRENCY (CAD)  <small>Please select</small>	

EDIT 

- ▼ react-app
 - > node_modules
 - ▼ public
 - ▼ styles
 - 🔗 main.scss
 - 🔗 variables.scss
 - <> index.html
 - ▼ src
 - ▼ components
 - ▼ main
 - ▼ home
 - ⚙️ HomePage.jsx
 - 🔗 HomePage.scss
 - > quotes
 - ▼ navbar
 - ⚙️ Navbar.jsx
 - 🔗 Navbar.scss
 - ▼ sidemenu
 - ⚙️ SideMenu.jsx
 - ⚙️ SideMenuBar.jsx
 - 🔗 SideMenuBar.scss
 - ⚙️ SideMenuComponent.jsx
 - ⚙️ SideMenuItem.jsx
 - # App.css
 - ⚙️ App.jsx
 - JS App.test.js
 - # index.css
 - JS index.js

BACK END (express-app)

1. Libraries

- [Express](#) for Back End
- [Body-parser](#), [chalk](#), [cors](#), [dotenv](#), [fs](#), [nodemon](#), [pinocolada](#) - helper libraris/middlewares
- [Express-validator](#) - used to validate data passed from React app. Additional validation, in case of middle-man attack
- [Nodemailer & Mailgen](#) - Sending quote to client Email, with html type format
- [PG](#) - for DB connection
-

WET BAT Agency

Hi Tima Juma,

Please, find your quote

Route	Dates	Price
Toronto to Calgary	2020-12-01 - 2020-12-02	\$ 1230

Looking forward to do more business with you

2. File structure

- **Server.js** – tuning
- **ApiRoutes.js** – handles all endpoints in format of “**api/endpoint**”
 - i. Uses functions from **dbFunctions.js** with **async/await**
 - ii. Uses middleware to validate form data
 - iii. Uses middleware to send quote to client email
- Further other types of endpoints can be utilized in same format “**user/endpoint**”, etc
- **Databse folder** contains sql files, of :
 - i. Schema/ Create
 - ii. Seeds/Seeds
 - iii. Queries/ SqlQuery – makes easy deployment
- **DbSetup.js** for DB setup, parameters of Elepehant Sql imported from .env file
- **dbFunstions** makes queries to DB, try/catch used for error handling.

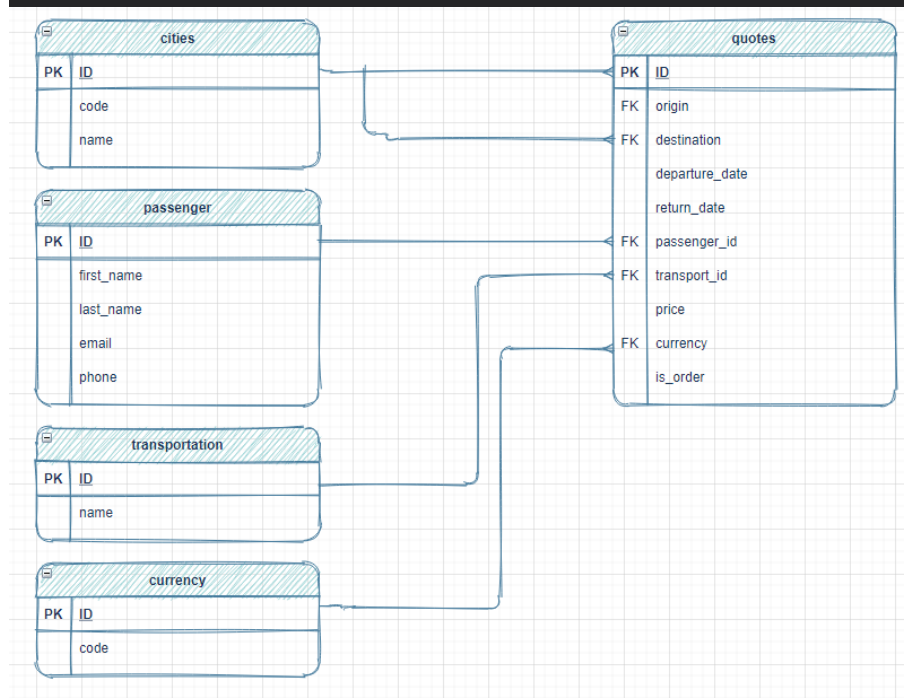
3. Functionality

- **Server.js** – creates express APP, passes db and router to Apiroutes
- Sensitive data stored in **.env** file. Which is not pushed to github
- All endpoint testing dome with **REST client**
- Database resets on command

```

  express-app
  |
  |> bin
  |> controllers
  |
  |< database
  |   |
  |   |< queries
  |   |   |
  |   |   |< selectAllquote.sql
  |   |   |
  |   |   |< schema
  |   |   |   |
  |   |   |   |< create.sql
  |   |   |   |
  |   |   |   |< seeds
  |   |   |   |   |
  |   |   |   |   |< seeds.sql
  |   |   |   |
  |   |   |   |< dbFunctions.js
  |   |   |   |< dbSetup.js
  |   |   |   |< resetdb.js
  |   |   |
  |   |   |< middlewares
  |   |   |   |
  |   |   |   |< formValidation.js
  |   |   |   |< sendEmail.js
  |   |   |
  |   |   |> node_modules
  |   |   |
  |   |   |< routes
  |   |   |   |
  |   |   |   |< apiRoutes.js
  |   |   |
  |   |   |< test
  |   |   |   |
  |   |   |   |< test.rest
  |   |   |
  |   |   |< .env
  |   |   |   |
  |   |   |   |< .env.example
  |   |   |
  |   |   |< package-lock.json
  |   |   |< package.json
  |   |   |
  |   |   |< server.js

```



SCALABILITY

1) Express:

- Structure is organized close to MVC design pattern, so as any additional routing and be added to `/routes` folder
- `/controller` folder can be used by passing to `/routes` folder
- New queries can be added to `dbFunction` file or new file based on route, only importing DB from `dbSetup.js`
- New middlewares can be added to `/middleware` file
- `/database` folder contains schema folder where new relations can be extended

2) React:

- All new components can be added to `/main` following many routes
- Material-UI makes solid design principle on components
- Sass allows easy control of color and other variables
- Formik makes easier dealing with forms on large scale