

# Modules

---

*App*

*Footer*

*FooterScrubBar*

*MediaControls*

*AlertBtn*

*DarkModeToggle*

*Datetime*

*Header*

*HistoryToggle*

*Logo*

*MenuBtn*

*Loading*

*BusIcon*

*Layers*

*MapWrapper*

*ColorPicker*

*Filter*

*InfoCard*

*MUITable*

*Sidetabs*

*Table*

*Theme*

*ScheduleHelper*

[api-access](#)[api-calls](#)[Controllers](#)[DataHelper](#)

## Constants

---

[processedData](#)

## App

---

- [App](#)
  - [~CreateRoutes\(data, setRoutesArray\)](#)
  - [~App\(\) ΓçÆ](#)
  - [~Hooks\(waypoints, availableHistoryDates, data, date, play, historyMode, schedule, activeBus, theme, routesArray, oldRoutesArray, colors\)](#)
  - [~playCallback\(e\)](#)
  - [~changeDate\(newDate\)](#)
  - [~activeCallBack\(job\\_id\)](#)
  - [~switchTheme\(\)](#)
  - [~changeColors\(key, color\)](#)
  - [~useEffect\(\)](#)
  - [~fetchHistory\(date\)](#)
  - [~useEffect\(\)](#)

### App~CreateRoutes(data, setRoutesArray)

Kind: inner method of [App](#)

Param	Type	Description
data	Object	Scheduled data that is fetched from database
setRoutesArray	callback	Call back function to update state of App

### App~App() ΓçÆ

Entry point for entire application - Is injected into the 'body' DOM element with a class of 'root'.

**Kind:** inner method of [App](#)

**Returns:** The entire app as JSX

**\*\*Author\*\*:** Mark Dodson **\*\*Author\*\*:** James Hawes **\*\*Author\*\*:** Jamie Garner **\*\*Author\*\*:** Joseph Ising

**App~Hooks(waypoints, availableHistoryDates, data, date, play, historyMode, schedule, activeBus, theme, routesArray, oldRoutesArray, colors)**

global state hooks

**Kind:** inner method of [App](#)

Param	Type	Description
waypoints	array	array of waypoints in format [lat, long]
availableHistoryDates	array	array of Dates
data	Object	raw Data fetched from database
date	date	global Date
play	boolean	True = Play, false = Paused
historyMode	boolean	Switch history mode on and off
schedule	array	array of objects containing job info
activeBus	Object	Single Job object
theme	boolean	true = dark, false = light
routesArray	array	array of routes
oldRoutesArray	array	array of old routes
colors	Object	Object containing a color for each status.

**App~playCallback(e)**

Callback to switch between play states

Kind: inner method of [App](#)

Param	Type	Description
e	boolean	switches between play and pause

## App~changeDate(newDate)

Callback function to set and update global date of app

Kind: inner method of [App](#)

Param	Type	Description
newDate	date	Use this date to update the global date tracked in the app

## App~activeCallback(job\_id)

Callback function to update which bus is being tracked as active

Kind: inner method of [App](#)

Param	Type	Description
job_id	number	Unique number to identify each job

## App~switchTheme()

Switches between true and false and sets the new theme to local storage

Kind: inner method of [App](#)

## App~changeColors(key, color)

Callback function to update colorscheme of app. New color scheme is set to local storage

Kind: inner method of [App](#)

Param	Type	Description
key	number	number between 0 and existingColorScheme.length to update correct color

Param	Type	Description
color	string	Hex code of new color to be set

## App~useEffect()

Checks if browser has existing theme and color scheme set and loads it up. Otherwise, loads a default theme.

Kind: inner method of [App](#)

## App~fetchHistory(date)

Takes a date and makes a post request to the server to get relevant history information.

Kind: inner method of [App](#)

Param	Type	Description
date	date	Date object

## App~useEffect()

On app load, makes a get request to Express server. Server queries database and responds with raw Scheduled data

Kind: inner method of [App](#)

### Example

```
{
  "job_id": 400,
  "vehicle_id": 93,
  "driver_id": "JOHNSTON",
  "description_of_job": null,
  "pickup_time": "2022-01-13T08:30:00.000Z",
  "pickup_point": "Genazzano College - Group 2",
  "pickup_latitude": "-37.808730",
  "pickup_longitude": "145.056010",
  "destination_time": "2022-01-13T10:20:00.000Z",
  "destination": "Wesley College Glen Waverley Campus",
  "destination_latitude": "-37.875200",
  "destination_longitude": "145.154830",
  "empty_run": null,
  "req_facilities": null,
```

```
"routing_info": null
}
```

## Footer

---

- **Footer**
  - `~action` : number
  - `~Footer(props)` ΓÇÆ Component
  - `~handleDirectionChange(dir)`

### Footer~action : number

Keep track of the direction of the scrub bar. Where -1 = rewind, 1 = fast-forward and 0 = nothing

Kind: inner property of [Footer](#)

### Footer~Footer(props) ΓÇÆ Component

Container for Media Controls and Scrub bar

Kind: inner method of [Footer](#)

Returns: Component - Returns renderable component

Param	Type	Description
props	props	Passes various variables and props further down to child Components

### Footer~handleDirectionChange(dir)

Update the state with the provided number

Kind: inner method of [Footer](#)

Param	Type	Description
dir	number	Number. either -1, 0 or 1.

## FooterScrubBar

- [FooterScrubBar](#)
  - [~marks](#)
  - [~value](#)
  - [~padZero\(string\)](#) ΓÇÆ string
  - [~valuetext\(value\)](#) ΓÇÆ string
  - [~scrubTimer\(value\)](#) ΓÇÆ string
  - [~FooterScrubBar\(play, historyMode, action, setDirection\)](#) ΓÇÆ Component
    - [~handleChange\(event, newValue\)](#)
  - [~getTimeAsMinutes\(date\)](#)
  - [~useeffect\(\)](#)

## FooterScrubBar~marks

Provides markers for the scrub bar, where total minutes (value) is mapped to a time in hours:minutes format (label)

Kind: inner property of [FooterScrubBar](#)

### Example

```
const marks = [
  {
    value: 120,
    label: '02:00',
  },
  {
    value: 300,
    label: '05:00',
  },
  {
    value: 480,
    label: '08:00',
  },
  {
    value: 660,
    label: '11:00',
  },
  ...
]
```

## FooterScrubBar~value

Track the state of the value

Kind: inner property of [FooterScrubBar](#)

## FooterScrubBar~padZero(string) ΓçÆ string

Prepends a time string with a leading zero

Kind: inner method of [FooterScrubBar](#)

Param	Type	Description
string	string	The time string that needs to be prepended with a 00

## FooterScrubBar~valuetext(value) ΓçÆ string

Converts a value (time in minutes 0 - 1440) to a time.

Kind: inner method of [FooterScrubBar](#)

Param	Type	Description
value	number	number between 0 - 1440 ie. Time in minutes

## FooterScrubBar~scrubTimer(value) ΓçÆ string

scrub bar function to display actual scrub bar min and max set time for day step is how often to set points marks are the labeling of regular intervals

Kind: inner method of [FooterScrubBar](#)

Returns: string - formatted time string in 24hour time

Param	Type	Description
value	number	A time in minutes converted to hh:mm

### Example

```
scrubTimer(420) // returns 07:00
scrubTimer(930) // returns 15:30
```

## FooterScrubBar~FooterScrubBar(play, historyMode, action, setDirection) ΓçÆ Component



Tracks state of the Scrub bar and interval and renders it

**Kind:** inner method of [FooterScrubBar](#)

**Returns:** Component - renderable Component

Param	Type	Description
play	boolean	Boolean to set inner state
historyMode	boolean	If true then history mode is active
action	number	current Direction of scrub bar
setDirection	callback	Callback to set the direction in parent compoenent (Header)

### FooterScrubBar~handleChange(event, newValue)

Updates the value state

**Kind:** inner method of [FooterScrubBar](#)

Param	Type	Description
event	e	
newValue	number	Value to be set

### FooterScrubBar~getTimeAsMinutes(date)

Convers a time of the day to minutes

**Kind:** inner method of [FooterScrubBar](#)

Param	Type	Description
date	date	Date object

### Example

```
return (date.getHours() * 60) + date.getMinutes();  
Thus, 2:30 am:  
= (2 * 60) + 30  
= 120 + 30  
= 150 minutes
```

## FooterScrubBar~useffect()

Checks props passed to component and calculates whether the scrub bar should be moving and tells it how to move.

**Kind:** inner method of [FooterScrubBar](#)

## MediaControls

---

- [MediaControls](#)
  - [~MediaControls\(handleCallback\)](#) ΓçÆ Component
  - [~setPaused\(paused\)](#)
  - [~handleChange\(\)](#)

## MediaControls~MediaControls(handleCallback) ΓçÆ Component

function for displaying mmedia controls practically deprecated at the moment as media buttons are actually called in "HistoryToggle.js"

**Kind:** inner method of [MediaControls](#)

**Returns:** Component - Renderable component

Param	Type	Description
handleCallback	callback	Callback function to track state of play

## MediaControls~setPaused(paused)

Handle state of play and paused

**Kind:** inner method of [MediaControls](#)

Param	Type
paused	boolean

## MediaControls~handleChange()

Extends the onClick for the play pause buttons

**Kind:** inner method of [MediaControls](#)

## AlertBtn

---

### AlertBtn~AlertBtn() ΓÇÆ Component

function for an alert button to display when alerts are available also allow clicking and displaying of said alerts

**Kind:** inner method of [AlertBtn](#)

**Returns:** Component - Dom Element with Alers

## DarkModeToggle

---

- [DarkModeToggle](#)
  - [~DarkModeToggle\(switchTheme, theme\)](#) ΓÇÆ Component
  - [~handleChange\(\)](#)

### DarkModeToggle~DarkModeToggle(switchTheme, theme) ΓÇÆ Component

**Kind:** inner method of [DarkModeToggle](#)

**Returns:** Component - A toggle

**Funtion:** A toggle that animates on toggle, and sets state theme with a callback

Param	Type	Description
switchTheme	callback	Callback function to switch theme
theme	boolean	Sets inner state according to this.

### DarkModeToggle~handleChange()

Extends the onChange method provided by react

**Kind:** inner method of [DarkModeToggle](#)

## Datetime

---

- [Datetime](#)
  - [~isSameDay\(d1, d2\)](#) ΓÇÆ boolean
  - [~Datetime\(date\)](#) ΓÇÆ Component

## Datetime~isSameDay(d1, d2) ΓÇÆ boolean

Takes two Date objects returns true if they are same day (time of day is irrelevant).

**Kind:** inner method of [Datetime](#)

**Returns:** boolean - If dates are same: True. Otherwise false.

Param	Type	Description
d1	date	First Date
d2	date	Second Date

## Datetime~Datetime(date) ΓÇÆ Component

Component that keeps track of live time.

**Kind:** inner method of [Datetime](#)

**Returns:** Component - Renderable component

Param	Type	Description
date	date	Inner state is set with this

## Header

---

### Header~Header(props)

Main Header script for header component, design to give a top AppBar return an appBar with toolbar, with divs with responsive resizing each div calls another script to be placed inside the initial div

**Kind:** inner method of [Header](#)

Param	Type	Description
props	props	Various variables and callbacks that are further passed down to the children Components

### Example

```
<AppBar className="Header">
  <Toolbar>
```

```

        <div className="col-8 col-sm-1 col-md-1 col-lg-1 col-xl-1">
            <Logo theme={this.props.theme} />
        </div>
        <div className="d-none d-sm-block col-sm-2 col-md-2 col-lg-2 col-xl-2"
align="center">
            <Datetime date={this.props.date}/>
        </div>
        <div className="d-none d-sm-block col-sm-2 col-md-2 col-lg-2 col-xl-
7">
            <HistoryToggle changeDate={this.props.changeDate}
availableHistoryDates={this.props.availableHistoryDates}/>
        </div>
        <div className="d-none d-sm-none d-md-none d-lg-block col-lg-1 col-xl-
1">
            <AlertBtn />
        </div>
        <div className="d-none d-sm-none d-md-none d-lg-block col-lg-1 col-xl-
1">
            <DarkModeToggle theme={this.props.theme} switchTheme=
{this.props.switchTheme} />
        </div>
    </Toolbar>
</AppBar>

```

## HistoryToggle

- HistoryToggle
  - ~HistoryToggle(changeDate, availableHistoryDates) ΓÇÆ Component
  - ~state\_hooks(startDate, checked)
  - ~handleChange(nextChecked)
  - ~onTrigger(d, reset)

### HistoryToggle~HistoryToggle(changeDate, availableHistoryDates) ΓÇÆ Component

When toggled, datepicker is displayed. If date is changed, sets Global date of App.

Kind: inner method of [HistoryToggle](#)

Returns: Component - Renderable component

Param	Type	Description
changeDate	callback	Callback that sets the the global state of App.

Param	Type	Description
availableHistoryDates	array	Array of dates used to populate the calendar.

## HistoryToggle~state\_hooks(startDate, checked)

Inner state variables

Kind: inner method of [HistoryToggle](#)

Param	Type	Description
startDate	date	Date that the datepicker defaults to
checked	boolean	Track state of toggle

## HistoryToggle~handleChange(nextChecked)

The logic behind the toggle.

Kind: inner method of [HistoryToggle](#)

Param	Type	Description
nextChecked	boolean	Boolean value that is used to set state.

## HistoryToggle~onTrigger(d, reset)

Event handling. When toggleState changes, this fires.

Kind: inner method of [HistoryToggle](#)

Param	Type	Description
d	date	New Date to set
reset	boolean	Flag. If true, date should be reset back to current day.

## Logo

---

### Logo~Logo(theme) ΓÇÆ Component

Render the Logo

**Kind:** inner method of [Logo](#)  
**Returns:** Component - Renderable Logo component

Param	Type	Description
theme	boolean	Conditionally render the logo based on theme.

## MenuBtn

---

*Deprecated*

### ~~MenuBtn~MenuBtn() ΓÇÆ Component~~

*Deprecated*

**Kind:** inner method of [MenuBtn](#)  
**Returns:** Component - Hamburger Menu icon

## Loading

---

### Loading~Loading([center]) ΓÇÆ Icon

A loading icon that adds suspense

**Kind:** inner method of [Loading](#)  
**Returns:** Icon - Loading icon that spins

Param	Type	Default	Description
[center]	boolean	false	Optional field to alter styling and center icon

## BusIcon

---

- [BusIcon](#)
  - [~Schedule](#)
  - [~BusIcon\(props\) ΓÇÆ Component](#)

### BusIcon~Schedule

Building the icons with a name and color.

Kind: inner property of `BusIcon`

### Example

```
var schedule = null;
var name = "";
var busColor;
switch (props.type) {
  // ADD IN A COLOUR
  case "predeparted":
    schedule = props.schedule.filter(
      (buses) => buses.status === "Pre Departed"
    );
    busColor = props.colors.predeparted;
    name = "predeparted";
    break;
  case "ontime":
    schedule = props.schedule.filter(
      (buses) => buses.status === "On Time"
    );
    busColor = props.colors.ontime;
    name = "ontime";
    break;
  case "delayed":
    schedule = props.schedule.filter(
      (buses) => buses.status === "Delayed"
    );
    busColor = props.colors.delayed;
    name = "delayed";
    break;
  case "completed":
    schedule = props.schedule.filter(
      (buses) => buses.status === "Completed"
    );
    busColor = props.colors.completed;
    name = "completed";
    break;
  default:
    schedule = props.schedule.filter(
      (buses) => buses.status === "Pre Departed"
    );
    busColor = props.colors.predeparted;
    name = "predeparted";
}
```

## BusIcon~BusIcon(props) ΓçÆ Component



Each entry in the schedule is mapped to a BusIcon

**Kind:** inner method of [BusIcon](#)

**Returns:** Component - Renderable Component

Param	Type	Description
props	props	Various Objects and variables used to setup each Bus Icon

## Layers

- [Layers](#)
  - [~Layers\(schedule, activeBus, colors, waypoints, routesArray, oldRoutesArray, time, tracking\)](#)
  - [~setPosition\(position\)](#)
  - [~MyMarker\(props\)](#) ΓÇÆ Component

**Layers~Layers(schedule, activeBus, colors, waypoints, routesArray, oldRoutesArray, time, tracking)**

**Kind:** inner method of [Layers](#)

Param	Type	Description
schedule	Object	Raw schedule data - JSON object
activeBus	Object	A single entry from schedule data w/ a status attached
colors	Object	Global color scheme that is used for layers
waypoints	array	Array of lat, long coords
routesArray	array	Array of routes
oldRoutesArray	array	Array of old routes
time	date	Current time selected by Scrub bar
tracking	array	Array of tracking objects.

**Layers~setPosition(position)**

Track position state

Kind: inner method of [Layers](#)

Param	Type	Description
position	array	Array of lat long coords

Example

```
const position = [144.040383, -37.405732]
```

## Layers~MyMarker(props) ΓçÆ Component

Access active bus Marker to force open Popup on selection

Kind: inner method of [Layers](#)

Returns: Component - A leaflet Marker component

Param	Type	Description
props	props	Props required to build the marker

## MapWrapper

---

- [MapWrapper](#)
  - [~defaultPosition](#) : Object
  - [~MapWrapper\(props\) ΓçÆ](#)

## MapWrapper~defaultPosition : Object

Default Position map is centered about

Kind: inner property of [MapWrapper](#)

Example

```
const defaultPosition = { lat: -37.813629, lng: 144.963058 };
```

## MapWrapper~MapWrapper(props) ΓçÆ

Component that wraps the Map, Layers and Event handling

**Kind:** inner method of [MapWrapper](#)

**Returns:** Component

Param	Description
props	Destructure into schedule, activeBus, waypoints, Routes and callbacks

## ColorPicker

### ColorPicker~ColorPicker(k, color, changeColors) ΓÇÆ Component

**Kind:** inner method of [ColorPicker](#)

**Returns:** Component - A color Picker containing 8 predefined colors.

4 seperate color pickers being rendered:

Param	Type	Description
k	number	Unique key to identify which color picker this is. Allows multiple to be rendered.
color	string	Hex code for current selected color
changeColors	callback	Callback to update global color scheme

## Filter

### Filter~Filter(filter, setFilter) ΓÇÆ Component

Return a simple input field. Updates the filter prop with every key typed

**Kind:** inner method of [Filter](#)

**Returns:** Component - A search bar

Param	Type	Description
filter	string	The value after being filtered
setFilter	callback	Sets the value in the parent method

## InfoCard

---

- [InfoCard](#)
  - [~getRuntime\(pickup, dest\)](#) ΓçÆ number
  - [~InfoCard\(info, colors\)](#) ΓçÆ Component

### InfoCard~getRuntime(pickup, dest) ΓçÆ number

Takes two dates and returns the difference in minutes

**Kind:** inner method of [InfoCard](#)

**Returns:** number - Minutes between two dates

Param	Type	Description
pickup	date	Pickup datetime
dest	date	Destination datetime

### InfoCard~InfoCard(info, colors) ΓçÆ Component

Takes a json data as info and outputs to a component with styling and tabularized data.

**Kind:** inner method of [InfoCard](#)

**Returns:** Component - Information with table and styled display

Param	Type	Description
info	Object	Schedule information
colors	string	Hex codes to style with

## MUITable

---

- [MUITable](#)
  - [~MUITable\(props\)](#) ΓçÆ Component
  - [~handleEvent\(i\)](#)

### MUITable~MUITable(props) ΓçÆ Component

Extends external MuiDatables package

**Kind:** inner method of [MUITable](#)

**Returns:** Component - Component containing interactive table with styling and additional features

Param	Type	Description
props	props	
state.selectedRow	number	Track state of table rows

## MUITable~handleEvent(i)

Sets selected row into the class's state

**Kind:** inner method of [MUITable](#)

Param	Type	Description
i	number	Index of selected row

## Sidetabs

---

- [Sidetabs](#)
  - [~Sidetabs\(props\)](#) ΓÇÆ Component
  - [~setOpenTab\(openTab\)](#)
  - [~onClose\(\)](#)
  - [~onOpen\(\)](#)

## Sidetabs~Sidetabs(props) ΓÇÆ Component

Extends the react-leaflet-sidetabs package

**Kind:** inner method of [Sidetabs](#)

**Returns:** Component - Floating sidebar

Param	Type	Description
props	props	Various components and variables

## Sidetabs~setOpenTab(openTab)

Track state of which tab is open

Kind: inner method of [Sidetabs](#)

Param	Type	Description
openTab	string	String that represents an internal route to each tab

## Sidetabs~onClose()

Event listener that updates state of tabs

Kind: inner method of [Sidetabs](#)

## Sidetabs~onOpen()

Event listener that updates state of tabs

Kind: inner method of [Sidetabs](#)

## Table

- [Table](#)
  - [~getTime\(value\)](#)  $\Gamma_{\zeta\mathcal{A}\mathcal{E}}$  Component
  - [~Table\(schedule, activeCallBack, activeBus\)](#)  $\Gamma_{\zeta\mathcal{A}\mathcal{E}}$  Component
  - [~handleEvent\(row, i\)](#)

## Table~getTime(value) $\Gamma_{\zeta\mathcal{A}\mathcal{E}}$ Component

Kind: inner method of [Table](#)

Returns: Component - Stringified local time in 24 hour format

Param	Type	Description
value	date	Datetime object

## Table~Table(schedule, activeCallBack, activeBus) $\Gamma_{\zeta\mathcal{A}\mathcal{E}}$ Component

Kind: inner method of [Table](#)

Returns: Component - A functional table with styling and sort/search features

Param	Type	Description
schedule	Object	Raw schedule data

Param	Type	Description
activeCallBack	callback	Update state of activeBus in parent component
activeBus	Object	An entry from the schedule that represents on vehicle

## Table~handleEvent(row, i)

Essentially an event handler that sets the activeBus of the row that is selected.

Kind: inner method of [Table](#)

Param	Type	Description
row	Object	Object containing schedule data from a particular entry
i	number	Index of row in Schedule

## Theme

---

- [Theme](#)
  - [~lightTheme](#)
  - [~darkTheme](#)
  - [~GlobalStyle\(\)](#)

## Theme~lightTheme

All the properties and colors that make up the styling for the light theme

Kind: inner constant of [Theme](#)

### Example

```
export const lightTheme = {
  appBar: '#0074d9',
  scrubBar: '#1976d2',
  warning: 'rgb(255, 244, 229)',
  info: 'rgb(229, 246, 253)',
  error: 'rgb(253, 237, 237)',
  success: 'rgb(237, 247, 237)',
  selectedRow: '#1976d2'
};
```

## Theme~darkTheme

All the properties and colors that make up the styling for the dark theme

Kind: inner constant of [Theme](#)

### Example

```
export const darkTheme = {
  body: '#2C2F33',
  text: '#FFF',
  background: '#2C2F33',
  appBar: '#23272A',
  icon: 'rgb(133, 184, 88)',
  scrubBar: 'rgb(133, 184, 88)',
  warning: 'rgb(255, 215, 157)',
  info: 'rgb(153, 186, 240)',
  error: 'rgb(247, 186, 186)',
  success: 'rgb(153, 240, 169)',
  mapTiles: 'brightness(0.6) invert(1) contrast(4) hue-rotate(220deg)
  saturate(0.4) brightness(0.4)',
  selectedRow: 'rgb(133, 184, 88)'
};
```

## Theme~GlobalStyle()

Mixes Javascript and CSS to create dynamic styling that is cascaded down the heirarchy of components whereby, each child component has access to the context.

Kind: inner method of [Theme](#)

## ScheduleHelper

---

- [ScheduleHelper](#)
  - *static*
    - [.calculatedSchedule](#)  $\Gamma_{\mathbb{A}}$  Object
  - *inner*
    - [~getStatus\(bus, currentTime, tracking\)](#)  $\Gamma_{\mathbb{A}}$  string

### ScheduleHelper.calculatedSchedule $\Gamma_{\mathbb{A}}$ Object

Takes a raw schedule and calculates a status for each bus and returns new schedule



**Kind:** static constant of [ScheduleHelper](#)

**Returns:** `Object` - With an appended status to each bus

Param	Type	Default	Description
<code>rawData</code>	<code>Object</code>		Raw schedule data
<code>time</code>	<code>date</code>		Datetime object
<code>[tracking]</code>	<code>Object</code>		Optional tracking history information to build an accurate status from.

## **ScheduleHelper~getStatus(bus, currentTime, tracking) ΓÇÆ string**

Takes a bus and time and returns an approximate status

**Kind:** inner method of [ScheduleHelper](#)

**Returns:** `string` - The calculated status of the bus

Param	Type	Description
<code>bus</code>	<code>Object</code>	Bus object
<code>currentTime</code>	<code>date</code>	Datetime object of current time
<code>tracking</code>	<code>Object</code>	Provided tracking history waypoints to fetch accurate status's. Handles null values.

## **api-access**

- [api-access](#)
  - [~getAPIKey\(\)](#) ΓÇÆ `string`
  - [~getGPSVehicles\(apikey\)](#) ΓÇÆ `array`
  - [~getGPSLocationHistory\(apikey, vehicleid, startdatetime, enddatetime\)](#) ΓÇÆ `array`
  - [~getCurrentGPSSnapshot\(apikey\)](#) ΓÇÆ `Object`
  - [~getScheduledVehicles\(apikey\)](#) ΓÇÆ `Object`
  - [~getScheduledActivity\(apikey, date\)](#) ΓÇÆ `Object`

## **api-access~getAPIKey() ΓÇÆ string**

Creates a post request to Nuline's server in accordance with provided API instructions.

**Kind:** inner method of [api-access](#)

**Returns:** string - API key in order to create a session

**Example**

```
export function getAPIKey() {
  axios
    .post('http://rest.nulinecharter.com.au:7920', {
      'Request': 'Login',
      'Username': process.env.APIUSERNAME,
      'Password': process.env.APIPASSWORD,
      'API': API_VERSION
    })
    .then(res => {
      console.log(res.Login)
      console.log(res.APIInUse)
      console.log(res.LatestAPIAvailable)
      console.log(res.SessionKey)
      return res.SessionKey
    })
    .catch(error => {
      console.log("This is the command we sent: ", error.Error)
      console.log("We received no session key")
      console.log("This is the reason for the error: ", error.Reason)
    })
}
```

## api-access~getGPSVehicles(apikey) ΓçÆ array

RETURNS BUS ID, BUS NAME, BUS REGO OF EACH BUS MAY NEED MULTIPLE CALLS IF RETURNED DATA IS TOO BIG

**Kind:** inner method of [api-access](#)

Param	Type	Description
apikey	string	Fetches ApiKey

## api-access~getGPSLocationHistory(apiKey, vehicleid, startdatetime, enddatetime) ΓçÆ array

RETURNS LOCATION HISTORY FOR A GIVEN BUS ID, START AND END DATETIMES FORMAT: yyyy-MM-ddThh:mm:ss

**Kind:** inner method of [api-access](#)

**Returns:** array - GPS location history

Param	Type	Description
apiKey	string	Key retrieved with getApiKey call
vehicleid	number	Unique identifier for a vehicle
startdatetime	date	Timestamp representing start time
enddatetime	date	Timestamp representing end time

### Example

```
const startdatetime = new Date(2021, 11, 25); // 25th December 2021
const enddatetime = new Date(2021, 11, 26); // 26th December 2021

const vehicleHistory = getGPSLocationHistory('asldffk2q44324laiejfdkt398df', 101,
startdatetime, enddatetime);
```

## api-access~getCurrentGPSSnapshot(apikey) ΓçÆ Object

RETURNS THE SNAPSHOT OF CURRENT GPS LOCATIONS (10 MINUTE INTERVALS)

**Kind:** inner method of [api-access](#)

Param	Type	Description
apikey	string	ApiKey

## api-access~getScheduledVehicles(apikey) ΓçÆ Object

RETURNS THE CURRENT SCHEDULE (LIST OF BUSSES CURRENTLY ASSIGNED TO THE SCHEDULE)

**Kind:** inner method of [api-access](#)

Param	Type	Description
apikey	string	ApiKey

## api-access~getScheduledActivity(apikey, date) ΓçÆ Object

RETURNS THE SCHEDULE ACTIVITY (COMPLETE LIST OF LOCATIONAL DATA, START/END DESTINATION COORDS+DESCRIPTIONS) //FOR A GIVEN DAY (yyyy-MM-dd)

Kind: inner method of [api-access](#)

Param	Type	Description
apikey	string	ApiKey
date	date	

## api-calls

---

- [api-calls](#)
  - [~GetGPSVehicles\(req, res\)](#) ΓÇÆ Object
  - [~GetScheduledActivity\(req, res\)](#) ΓÇÆ Object

### api-calls~GetGPSVehicles(req, res) ΓÇÆ Object

queries database for Vehicle data and returns to client

Kind: inner method of [api-calls](#)

Returns: Object - Http response with data and a status code attached

Param	Type	Description
req	Object	Http request
res	Object	Http response

### api-calls~GetScheduledActivity(req, res) ΓÇÆ Object

queries database for scheduled activity data and returns to client

Kind: inner method of [api-calls](#)

Returns: Object - Http response with data and a status code attached

Param	Type	Description
req	Object	Http request
res	Object	Http response

# Controllers

---

- [Controllers](#)
  - [~getAllAddress\(req, res\)](#)  $\Gamma_{\zeta\mathcal{A}} \text{ Object}$
  - [~getHistory\(req, res\)](#)  $\Gamma_{\zeta\mathcal{A}} \text{ Object}$
  - [~getAddressID\(req, res\)](#)  $\Gamma_{\zeta\mathcal{A}} \text{ Object}$
  - [~getSchedule\(req, res\)](#)  $\Gamma_{\zeta\mathcal{A}} \text{ Object}$

## Controllers~getAllAddress(req, res) $\Gamma_{\zeta\mathcal{A}} \text{ Object}$

queries database for address data and returns to client

**Kind:** inner method of [Controllers](#)

**Returns:** `Object` - Http response with data and a status code attached

Param	Type	Description
req	Object	Http request
res	Object	Http response

## Controllers~getHistory(req, res) $\Gamma_{\zeta\mathcal{A}} \text{ Object}$

queries database for history data and returns to client

**Kind:** inner method of [Controllers](#)

**Returns:** `Object` - Http response with data and a status code attached

Param	Type	Description
req	Object	Http request containing a specific date.
res	Object	Http response

## Controllers~getAddressID(req, res) $\Gamma_{\zeta\mathcal{A}} \text{ Object}$

queries database for a specific address and returns to client

**Kind:** inner method of [Controllers](#)

**Returns:** `Object` - Http response with data and a status code attached

Param	Type	Description
-------	------	-------------

Param	Type	Description
req	Object	Http request containing an address id
res	Object	Http response

## Controllers~getSchedule(req, res) ΓÇÆ Object

queries database for schedule data and returns to client

Kind: inner method of [Controllers](#)

Returns: Object - Http response with data and a status code attached

Param	Type	Description
req	Object	Http request
res	Object	Http response

## DataHelper

---

- [DataHelper](#)
  - [~FILE](#) : string
  - [~writeScheduleToFile\(schedule\)](#)
  - [~readScheduleFromFile\(\)](#) ΓÇÆ Object
  - [~getScheduledActivity\(\)](#) ΓÇÆ Object

### DataHelper~FILE : string

filepath of temp file

Kind: inner constant of [DataHelper](#)

### DataHelper~writeScheduleToFile(schedule)

Caches the day's schedule to disk. Writes the raw schedule to a .JSON file

Kind: inner method of [DataHelper](#)

Param	Type	Description
schedule	array	Array of objects (JSON data)

## DataHelper~readScheduleFromFile() ΓçÆ Object

Reads the cached schedule from file

Kind: inner method of [DataHelper](#)

Returns: Object - JSON object of day's schedule

## DataHelper~getScheduledActivity() ΓçÆ Object

Reads the cached schedule from file

Kind: inner method of [DataHelper](#)

Returns: Object - JSON object of day's schedule

## ~~processedData~~

---

*Deprecated*

Kind: global constant