**Travella – Travel Log Web Application**

# **Overview**

**Travella is a travel log web application giving its user the capability to save their travels, hikes, visits and adventure in a shareable form for others to view.**

Others can see where the user has been, his thoughts/comments on the place, ratings and image. We aim to use a map to show all these functionalities, so the user has access to a map for directions - which also acts as a record/log and at the same time they can share with friends and family who are not with them. It can be accessed on most devices and platforms – a web browser is all the client needs to be able to use and view it.

This document mainly discusses the interactions between the user and the application and not much about the goings on in the backend of the application.

# **User Interface**

The Travella is an application with an interface made up mostly of a map. It will have markers and popups will be visible when the markers are clicked on. The popups will be used to display information on the map but only on places where there are marker and not until clicked.

A brighter map style is used in order to improve visibility of the map. The images are optimized to be able to fit into a popup height of 200px and a width of 250px for a good and optimized UI.

The map will have a create new log entry feature for the client to add new information to the map. The data input will be sent to a database and rendered on the application from the database.

Map controls like Geo-location, Fullscreen and Navigation control are added to the map to improve User Interface (UI) and User Experience (UX).

# **Functionality**

Travella is a travel log map application that provides a database, map and log form. It populates latitude, longitude, image and date from the database into the map. The user also has the choice of entering new travel log information through a form which validates that the requirements of the database are met, stored in the database and sent back to the front-end all-in milliseconds.

The possible failures are:

* Client IP address blocking: this is solved by using the “*trust proxy*” function in the server.
* Requests to the Mapbox API may exceed amount allowed for a free tier: there is a choice for the user to go for a pro Mapbox API account.
* The user may make entries can harm the database: checks have been put in the form to make sure that only what the database needs are allowed to pass through the form and nothing more.
* GeoLocation Control– the map will be able to access GPS & network features of the device to get accurate user location.
* Fullscreen Control – the user can control the importance of the map on the screen.
* Navigation Control – a compass is to show true north every time your map moves.
* Secure log entry with a Log\_Key: after deployment the log entry form of the Travella app is secured from unauthorized entry.

There are no limitations as to the number of new log entries a user can make on the Travella app. The users are only limited by their wander lust and desire to share their journey/adventure with others.

# **Statement of Goals**

This application is intended for those who have the desire to visit places, the passion for adventure and the need to share with friends and families the beautiful and sometimes daring places they have visited.

The intention of this app is that it can be used as a map and a travel log. Any other reasons the users find for the app within legal boundaries is welcome and encouraged so long as the user bears responsibility for such uses.

# **Milestones**

### **Setup Server**

1. Install Dependencies
2. Install/Setup Linter
3. Setup Express App
4. Setup Not Found and Error Middlewares

### **Model Database**

1. Setup Mongoose Model(s)
2. POST/logs

### **Create a new log entry**

1. GET/logs
2. Enter Log\_Key
3. List all log entries

## **Setup Client**

### **Create a Form to add new Entry**

1. Create react app.
2. Create Log Entry Form.

### **Setup Map SDK on client**

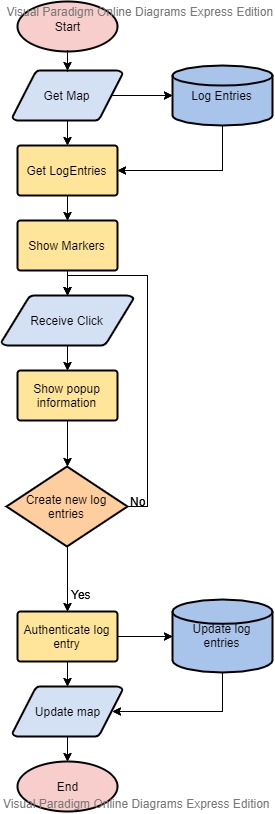
1. Get MapBox Account.
2. Create Map Styles with MapBox Studio.
3. Install ReactMapGL.

### **List all log entries on map**

1. Connect to Server.
2. List all log entries on map using markers and popups.

## **DEPLOY!**

# **Travella – Data Flow Chart**



# **Travella Modified Source Code**

### **App.js**

import React, {useState, useEffect} from 'react';

import ReactMapGL, {Marker, Popup, ScaleControl, NavigationControl, FullscreenControl, GeolocateControl} from 'react-map-gl';

import {listLogEntries} from './API';

import LogEntryForm from './LogEntryForm';

const MAPBOX\_TOKEN = 'pk.eyJ1IjoiaWFtcHJpbWUiLCJhIjoiY2thZzc5MXcwMDNybjJzb3lxZnFqcWIxOSJ9.HKCegM3Kueaf5RxItCZqUQ';

const MAP\_STYLE = "mapbox://styles/iamprime/ck75rb08g0iry1ik96iif7jkl";

const fullscreenCtrlStyle = {

  position: 'absolute',

  top: 0,

  left: 0,

  padding: '10px'

};

const navStyle = {

  position: 'absolute',

  top: 36,

  left: 0,

  padding: '10px'

};

const scaleControlStyle = {

  position: 'absolute',

  bottom: 36,

  left: 0,

  padding: '10px'

};

const geoLocateCtrl = {

  position: 'absolute',

  top: 130,

  left: 0,

  padding: '10px'

}

const App = () => {

  const [logEntries, setLogentries] = useState([]);

  const [showPopup, setShowPopup] = useState({});

  const [addEntryLocation, setAddEntryLocation] = useState(null);

  const [viewport, setViewport] = useState({

    width: '100vw',

    height: '100vh',

    latitude: 5.57205,

    longitude: 7.058823,

    zoom: 3,

    maxZoom: 20

  });

  const getEntries = async () => {

    const logEntries = await listLogEntries();

    setLogentries(logEntries);

  };

  useEffect(() => {

    getEntries();

  }, []);

  const showAddMarkerPopup = (event) => {

    const [longitude, latitude] = event.lngLat;

    setAddEntryLocation({

      latitude,

      longitude,

    });

  }

  return (

    <ReactMapGL

      {...viewport}

      mapStyle={MAP\_STYLE}

      mapboxApiAccessToken={MAPBOX\_TOKEN}

      onViewportChange={setViewport}

      onClick={showAddMarkerPopup}

    >

      {

        logEntries.map(entry => (

          <React.Fragment key={entry.\_id}>

          <Marker

            latitude={entry.latitude}

            longitude={entry.longitude}

            offsetLeft={-15}

            offsetTop={-30}

          >

            <div

              onClick={() => setShowPopup({

                [entry.\_id]: true,

              })}

            >

              <svg

              viewBox="0 0 24 24"

              width="30px"

              height="30px"

              stroke="darkmagenta"

              strokeWidth="2"

              fill="gold"

              strokeLinecap="round"

              strokeLinejoin="round"

              >

                <path d="M21 10c0 7-9 13-9  13s-9-6-9-13a9 9 0 0 1 18 0z"></ path><circle cx="12" cy="10" r="3"></  circle>

              </svg>

            </div>

        </Marker>

        {

          showPopup[entry.\_id] ? (

            <Popup

              latitude={entry.latitude}

              longitude={entry.longitude}

              dynamicPosition={true}

              sortByDepth={true}

              closeButton={true}

              closeOnClick={false}

              onClose={() => setShowPopup({

              })}

              anchor="top" >

              <div className="popup">

                <h3>{entry.title}</h3>

                {entry.image ? <img src={entry.image} alt={entry.title}/> : null}

                <p>{entry.description}</p>

                <p>{entry.comments}</p>

                <small>Experience Rating: {entry.rating} <br/>

                Visited on: {new Date(entry.visitDate).toLocaleDateString()}</small>

              </div>

            </Popup>

          ) : null

        }

        <div style={fullscreenCtrlStyle}>

          <FullscreenControl/>

        </div>

        <div style={navStyle}>

          <NavigationControl/>

        </div>

        <div style={scaleControlStyle}>

          <ScaleControl/>

        </div>

        <div style={geoLocateCtrl}>

          <GeolocateControl

positionOptions={{enableHighAccuracy: true}}

             trackUserLocation={true}

             showUserLocation={true}

/>

        </div>

      </React.Fragment>

        ))

      }

      {/\*\* Add New Travella Information \*/}

      {

        addEntryLocation ? (

          <>

            <Marker

              latitude={addEntryLocation.latitude}

              longitude={addEntryLocation.longitude}

            >

              <div>

                <svg

                viewBox="0 0 24 24"

                width="30px"

                height="30px"

                stroke="midnightblue"

                strokeWidth="2"

                fill="goldenrod"

                strokeLinecap="round"

                strokeLinejoin="round"

                >

                  <path d="M21 10c0 7-9 13-9    13s-9-6-9-13a9 9 0 0 1 18 0z"></  path><circle cx="12" cy="10"   r="3"></  circle>

                </svg>

              </div>

            </Marker>

            <Popup

              latitude={addEntryLocation.latitude}

              longitude={addEntryLocation.longitude}

              dynamicPosition={true}

              closeButton={true}

              closeOnClick={false}

              onClose={() => setAddEntryLocation(null)}

              anchor="top" >

              <div className="popup">

                <LogEntryForm

                  onClose={() => {

                    setAddEntryLocation(null);

                    getEntries();

                  }}

                  location={addEntryLocation}

                />

              </div>

          </Popup>

          </>

        ) : null

      }

    </ReactMapGL>

  );

}

export default App;