Assignment 03
CS-4032 Web Programming Fall 2022

Build a Bus Tracker API using Node.js, Express.js and Mongo

Open Date - Nov 03, 2022 **Deadline - Nov 13, 2022 11:30 PM**

Instructions

- Create a bus tracker API similar to Chicago Transit Authority Bus Tracker API **using Node.js, Express.js and Mongo**
- Chicago Transit Authority Bus Tracker API documentation is provided along with this assignment API.
- You are not allowed to copy/paste code from the internet as it is.Zero for copying the code from the source/internet.
- 70% of the code must be written by yourself.

Overview

Chicago Transit Authority Bus Tracker API Bus Tracker API allows for querying data from the CTA Bus Tracker service, with an XML document/JSON as output. The system can provide estimated arrival times for buses as they approach bus stops, as well as route, service and vehicle location data. Information about buses and their estimated arrivals is updated about once every minute. Data available through the API includes:

- Time
- Vehicle
- Routes
- Route Directions
- Stops
- Patterns
- Prediction
- Error Codes

How Bus Tracker works

Each bus has a device that uses the Global Positioning System (GPS) to determine where it is, make announcements as buses approach stops, and report back information about the bus to our servers. It helps us see where buses are and make service and schedules better and it can help bus riders make decisions about when to head out to a bus stop.

How Bus Tracker works

The developer API uses the same data from the BusTime system, which powers CTA Bus Tracker. Information about the location, direction and status of CTA buses is fed from each bus and delivered to the BusTime system, which then can show where buses are or estimate arrival times to stops ahead of a bus.Data is updated about once per minute, and arrival estimations are based on how long it normally takes for a bus to get from one place to the next. Because traffic conditions and other unexpected delays occur, we can't predict precisely when a bus will arrive—only estimate based on normal travel times during the time of day where an estimate is occurring.

In order to use the API, the user must have valid API key The key allows the user to make calls to the API as it is included in every data request.You can use the below key to call the API methods:

Key :ujAhaYu9dy6TAF2VgMLWK5nnV

For example, to request the current system time through the developer API, a program or script will make a HTTP/1.1 GET request to the following URL with parameters:

Get System Time:
https://www.ctabustracker.com/bustime/api/v2/gettime?key=ujAhaYu9dy6TAF2VgMLWK5nnV&format=json

GetVehicles:
https://ctabustracker.com/bustime/api/v2/getvehicles?key=ujAhaYu9dy6TAF2VgMLWK5nnV&rt=20&format=json

Get Routes :
http://ctabustracker.com/bustime/api/v2/getroutes?key=ujAhaYu9dy6TAF2VgMLWK5nnV&format=json

Get Directions:
http://ctabustracker.com/bustime/api/v2/getdirections?key=ujAhaYu9dy6TAF2VgMLWK5nnV&rt=20&format=json

Get Stops:
https://ctabustracker.com/bustime/api/v2/getstops?key=ujAhaYu9dy6TAF2VgMLWK5nnV&rt=7&dir=Eastbound&format=json

GetPatterns:
https://ctabustracker.com/bustime/api/v2/getpatterns?key=ujAhaYu9dy6TAF2VgMLWK5nnV&rt=20&pid=954&format=json

Tasks:

1 - Call above API methods from your NodeExpress Server to get the data of Vehicles, directions, Routes, stops and patterns.
2- Make schema of each entity as models using mongoose
3- Save the fetched data into Mongodb collections accordingly
5- Perform CRUD on each entity :Vehicles, directions, Routes, stops and patterns.
6- Add a separate route on each CRUD operation so that it can be tested by the postman.

Note: you can get the schema understanding of each entity in provided API documentation.

For case of any query, you can write an email at
sidra.khalid@nu.edu.pk