

# Planning Document

## Dataset and Task

The dataset consists of multiple txt files, each representing different aspects of the TransLink public transport system in South East Queensland. The key tables include:

- **calendar\_dates.txt**: Contains special service dates and any exceptions to the regular schedule, identified by service\_id, date, and exception\_type.
- **calendar.txt**: Defines the operational schedule of services, with columns representing the days of the week (monday to sunday) and the period of service (start\_date, end\_date).
- **stops.txt**: Lists all bus stops, including details such as stop\_id, stop\_name, stop\_lat, and stop\_lon, which provide the stop's identifier and its geographical location.
- **routes.txt**: Contains information on different bus routes, such as route\_id, route\_short\_name, and route\_long\_name, providing a descriptive overview of each route.
- **trips.txt**: Specifies the trips made by the vehicles, linked to the route\_id and service\_id, and includes details like trip\_id and trip\_headsign.
- **stop\_times.txt**: Provides the schedule times for stops, detailing arrival\_time, departure\_time, and the sequence of stops (stop\_sequence) for each trip.

In addition to the static data files, there are three JSON files that represent dynamic data fetched from a live API, providing real-time updates on public transport:

- **trip\_updates.json**: Contains real-time information on trip updates, including delays, changes in schedules, and other trip-related information.
- **vehicle\_position.json**: Provides live data on the current position of vehicles, including their latitude, longitude, and other movement-related data.
- **alerts.json**: Includes real-time alerts about the service, such as disruptions, cancellations, or important notices affecting the transport services.

These JSON files are used to enhance the static data with real-time updates, enabling the application to provide accurate and timely information to users.

**Task:** Create a terminal-based Node.js application that parses TransLink data and provides a summary of all bus route stops in South East Queensland. The application must be built using functional programming principles and include features like data filtering, fetching, and caching.

<div>calendar_dates</div> <table><tr><td>service_id</td><td>varchar</td></tr><tr><td>date</td><td>date</td></tr><tr><td>exception_type</td><td>integer</td></tr></table>	service_id	varchar	date	date	exception_type	integer	<div>calendar</div> <table><tr><td>service_id</td><td>varchar</td></tr><tr><td>monday</td><td>integer</td></tr><tr><td>tuesday</td><td>integer</td></tr><tr><td>wednesday</td><td>integer</td></tr><tr><td>thursday</td><td>integer</td></tr><tr><td>friday</td><td>integer</td></tr><tr><td>saturday</td><td>integer</td></tr><tr><td>sunday</td><td>integer</td></tr><tr><td>start_date</td><td>date</td></tr><tr><td>end_date</td><td>date</td></tr></table>	service_id	varchar	monday	integer	tuesday	integer	wednesday	integer	thursday	integer	friday	integer	saturday	integer	sunday	integer	start_date	date	end_date	date	<div>routes</div> <table><tr><td>route_id</td><td>varchar</td></tr><tr><td>route_short_name</td><td>varchar</td></tr><tr><td>route_long_name</td><td>varchar</td></tr><tr><td>route_desc</td><td>text</td></tr><tr><td>route_type</td><td>integer</td></tr><tr><td>route_url</td><td>varchar</td></tr><tr><td>route_color</td><td>varchar</td></tr><tr><td>route_text_color</td><td>varchar</td></tr></table>	route_id	varchar	route_short_name	varchar	route_long_name	varchar	route_desc	text	route_type	integer	route_url	varchar	route_color	varchar	route_text_color	varchar	<div>stop_times</div> <table><tr><td>trip_id</td><td>varchar</td></tr><tr><td>arrival_time</td><td>time</td></tr><tr><td>departure_time</td><td>time</td></tr><tr><td>stop_id</td><td>varchar</td></tr><tr><td>stop_sequence</td><td>integer</td></tr><tr><td>pickup_type</td><td>integer</td></tr><tr><td>drop_off_type</td><td>integer</td></tr></table>	trip_id	varchar	arrival_time	time	departure_time	time	stop_id	varchar	stop_sequence	integer	pickup_type	integer	drop_off_type	integer
service_id	varchar																																																										
date	date																																																										
exception_type	integer																																																										
service_id	varchar																																																										
monday	integer																																																										
tuesday	integer																																																										
wednesday	integer																																																										
thursday	integer																																																										
friday	integer																																																										
saturday	integer																																																										
sunday	integer																																																										
start_date	date																																																										
end_date	date																																																										
route_id	varchar																																																										
route_short_name	varchar																																																										
route_long_name	varchar																																																										
route_desc	text																																																										
route_type	integer																																																										
route_url	varchar																																																										
route_color	varchar																																																										
route_text_color	varchar																																																										
trip_id	varchar																																																										
arrival_time	time																																																										
departure_time	time																																																										
stop_id	varchar																																																										
stop_sequence	integer																																																										
pickup_type	integer																																																										
drop_off_type	integer																																																										
<div>stops</div> <table><tr><td>stop_id</td><td>varchar</td></tr><tr><td>stop_code</td><td>varchar</td></tr><tr><td>stop_name</td><td>varchar</td></tr><tr><td>stop_desc</td><td>text</td></tr><tr><td>stop_lat</td><td>decimal</td></tr><tr><td>stop_lon</td><td>decimal</td></tr><tr><td>zone_id</td><td>varchar</td></tr><tr><td>stop_url</td><td>varchar</td></tr><tr><td>location_type</td><td>integer</td></tr><tr><td>parent_station</td><td>varchar</td></tr><tr><td>platform_code</td><td>varchar</td></tr></table>	stop_id	varchar	stop_code	varchar	stop_name	varchar	stop_desc	text	stop_lat	decimal	stop_lon	decimal	zone_id	varchar	stop_url	varchar	location_type	integer	parent_station	varchar	platform_code	varchar	<div>trips</div> <table><tr><td>route_id</td><td>varchar</td></tr><tr><td>service_id</td><td>varchar</td></tr><tr><td>trip_id</td><td>varchar</td></tr><tr><td>trip_headsign</td><td>varchar</td></tr><tr><td>direction_id</td><td>integer</td></tr><tr><td>block_id</td><td>varchar</td></tr><tr><td>shape_id</td><td>varchar</td></tr></table>	route_id	varchar	service_id	varchar	trip_id	varchar	trip_headsign	varchar	direction_id	integer	block_id	varchar	shape_id	varchar																						
stop_id	varchar																																																										
stop_code	varchar																																																										
stop_name	varchar																																																										
stop_desc	text																																																										
stop_lat	decimal																																																										
stop_lon	decimal																																																										
zone_id	varchar																																																										
stop_url	varchar																																																										
location_type	integer																																																										
parent_station	varchar																																																										
platform_code	varchar																																																										
route_id	varchar																																																										
service_id	varchar																																																										
trip_id	varchar																																																										
trip_headsign	varchar																																																										
direction_id	integer																																																										
block_id	varchar																																																										
shape_id	varchar																																																										

Figure 1. Static-Data Table

## Important Fields and Joins

- **route\_short\_name**: Key field for checking the existence of a specific route in the routes.txt dataset.
- **scheduled\_arrival\_time**: Field used to filter buses arriving within a specific time frame.
- **dateInput**: Field used to validate the correct date format.
- **timeInput**: Field used to validate the correct time format.
- **currentTime**: Reference time used to compare against scheduled\_arrival\_time for filtering upcoming buses.

## Steps to Perform the Task

1. **Load Route Data**: Load the route data from the routes.txt file.
  - **Action**: Use loadCSV to load the dataset containing routes.
2. **Validate Route Number**: Check if the provided route number exists in the loaded route data.
  - **Action**: Use some method to search for the route\_short\_name in the loaded data.
3. **Validate Date and Time Formats**: Ensure that the input date and time follow the correct formats (YYYY-MM-DD for date, HH:MM for time).
  - **Action**: Use regular expressions to match the date and time formats.
4. **Run Route Planner**: Execute the route planner function to retrieve the scheduled bus arrivals.
  - **Action**: Use runRoutePlanner to obtain bus arrival data.
5. **Filter Buses by Timeframe**: Filter the buses that are scheduled to arrive within 10 minutes of the current time.
  - **Action**: Compare scheduled\_arrival\_time from the bus data with currentTime to filter relevant records.
6. **Check Filtered Data**: Ensure that at least one bus meets the arrival timeframe criteria.
  - **Action**: Use an assertion to check that the length of the filtered data is greater than zero.

## Reference

ChatGPT was use to convert the static data file to the DBML format.