Setup:

1. This project was built using Rails 5.0.7.2 and Ruby 2.3.4
2. Once you are done cloning the project and running "bundle install", please make sure to run "rake db:seed". The function call in the seeds.rb file refreshes the data by deleting old records (if any), and parsing the CSV files. It then populates the database with new data. This data forms the backbone of the APIs/application, so running "rake db:seed" is required.

Assumptions:

1. I have assumed that a line is representative of a vehicle. Therefore, vehicle and line have been interchangeably used in some places.
2. I have assumed that delay values will get added to the arrival/departure times in times.csv.
3. The delays.csv file associates line_name with delay. Therefore, I am assuming that line/vehicle names are unique. If they were not, then (in the case of repeating line_names) we would not be able to pinpoint which specific line/vehicle the delay is associated with.
4. My solution assumes that the program will only be asked for vehicles at stop locations (given as x and y coordinates in stops.csv).
5. In order to find next vehicle at a given stop (problem 3), my API also needs the user to pass a time-value that is used as a "start time" to find the next vehicle at that stop. If there are no vehicles after the passed in "time-value", then the first vehicle is returned and as I'm assuming the same schedule will repeat everyday . The time-value here is passed in the same "HH:MM:SS" format as problem 1.
6. I assumed (and ensure through validations) that the location of every stop (combination of x and y) is unique
7. A TimeModel record (a row of the times.csv file) can belong to a single stop and line.

Design Choices:

1. I chose to use the database instead directly using csv data, so I could take advantage of ActiveRecord features, Rails validations (through models), and associations on model objects. Additionally, this would also be a more efficient choice for larger amounts (higher scale) of data.
2. I allow the user to refresh all the data through "rake db:seed" as that gives the user flexibility to alter CSV values and then add the new data to the database
3. I have an extremely light controller (ApiController) whose only responsibility is to call on a service and render a JSON response.
4. The *VerspaetungTransportationService* does most of the heavy-lifting of executing business logic in order to determine the response body for a request.
5. I do some basic type validation through the validation_service and have a simple heuristic for time-values that we get from the csv
6. My Api responses generally contain a "message" and "data" field. The "message" field gives some about the result and the "data" field gives the result itself
7. The CSVParser goes through all the csv files and updates the database with fresh data